NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A DESIGN OF A HARD DISK INTERFACE FOR
THE MICROPOLIS 1223-1

by

William Harold Brown

December 1981

Thesis Advisor: M. L. Cotton

Special Distribution

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Design of a Hard Disk Interface for the Micropolis 1223-1		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; December 1981
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) William Harold Brown		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1981
		13. NUMBER OF PAGES 87
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Special Distribution

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Hard Disk Interface, Micropolis Winchester Disk 1223-1

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis develops an interface to the Micropolis 8 inch Winchester disk drive model 1223-1, using an Intel 80/20 single board computer as the programmed input output device. This system is part of the AEGIS modeling group at the Naval Postgraduate School.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Special Distribution

A Design of a Hard Disk Interface for the Micropolis 1223-1

by

William H. Brown
Lieutenant, United States Navy
B.S. Physics Auburn University, 1975

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1981

ABSTRACT

This thesis develops an interface to the Micropolis 8 inch Winchester disk drive model 1223-1, using an Intel 80/20 single board computer as the programmed input output device. This system is part of the AEGIS modeling group at the Naval Postgraduate School.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Acknowledgements

I wish to express my sincere appreciation to my advisors and the members of the AEGIS group. The assistance they provided was invaluable in the preparation and writing of this report. I would also like to thank my brillant and beautiful wife without whom I would be nothing. She always comforts and consoles, never complains or interferes and writes my acknowledgements.

I. INTRODUCTION

A. PURPOSE OF THIS THESIS

The interface formulated for the hardware described herein was developed to provide a Intel 80/20 single board computer controller for the Micropolis 1223-1 hard disk unit. This system along with the interface will be available for the ongoing AEGIS modeling project at the Naval Postgraduate School. Furthermore the experience of wiring and programming a disk interface with a single board computer gave the author an opportunity to learn first hand about microcomputer hardware and programming techniques required for such a project.

B. ORGANIZATION OF THIS THESIS

This thesis is organized into descriptions of the hardware involved and the software required for a working Winchester disk interface. Additional attention is paid to the modification of an operating system to accomodate the Micropolis hard disk drive. Chapter 2 is a brief introduction into disk drives such as the Micropolis 1223-1. The operating characteristics, bus protocol and interface requirements are discussed in detail. Chapter 3 is a

discription of the Intel 80/20 single board computer and it's interface capabilities, followed by a discussion of the Intel Microcomputer Developement System (MDS) and it's role in the interface construction. Chapter 4 covers the actual interface design used including modifications of the hardware and software to meet the bus protocol requirements for successful communications with the Winchester disk. this chapter will conclude with some recommendations concerning the implementation of the disk into the AEGIS modeling project. Chapter 5 pertains to some of the difficulties encountered and recommendations for future applications of the system in the AEGIS model. The appendices contain the programs developed as part of this thesis for initialization and verification of the disk, and a read/write routine.

II. THE MICROPOLIS 1223-1 WINCHESTER DISK

A. OVERVIEW

High performance,high quality,and large capacity hard disk drives are now a low cost reality for microcomputer systems. Most hard disks use Winchester media, head technology, and other modern techniques to achieve high density and high performance. The bottom line specifications for high volume storage units are cost, reliability, capacity, and data access time.

One of the most attractive reasons for using a Winchester disk over a floppy disk system is that of dramatically increased capacity. Whereas a typical double sided double density floppy disk stores a maximum of 1.6 Megabytes of data, the average midrange Winchester can hold almost 18 Megabytes. Accessing data on an 8-inch Winchester disk takes an average of 48.2 microseconds. Compare that with about 100 microseconds for a double density floppy disk. With a Winchester disk dirt, fingerprints, scratches, and medium surface interferences are almost nonexistant. Winchester units are completely sealed after having been manufactured under cleanroom conditions.

B. THE MICROPOLIS 8-INCH DISK DRIVE

The model 1223-1 consist of a Micropolis fixed disk drive with an integral controller board. The 1223-1 has the same overall dimensions as an industry standard 8 inch flexible disk drive, has compatable mounting and requires the same D.C. supply voltages. The controller provides full data transfer and control facilities in six standard sectoring arrangements and can be attached to the host computer through a simple bus-oriented interface.

The Micropolis model selected for the AEGIS modeling group has 3 disk with 5 data surfaces, 580 tracks per data surface and a formatted capacity of 35.6 Megabytes. Each disk has been preset at the manufacturer for 24 sectors at 512 bytes each sector. The controller has a single sector buffer mode for asynchronous transfers between host and controller. Full error checking and error recovery procedures are automatically performed. Error correction code (ECC) is provided to ensure high integrity. A specification summary can be seen in Table I.

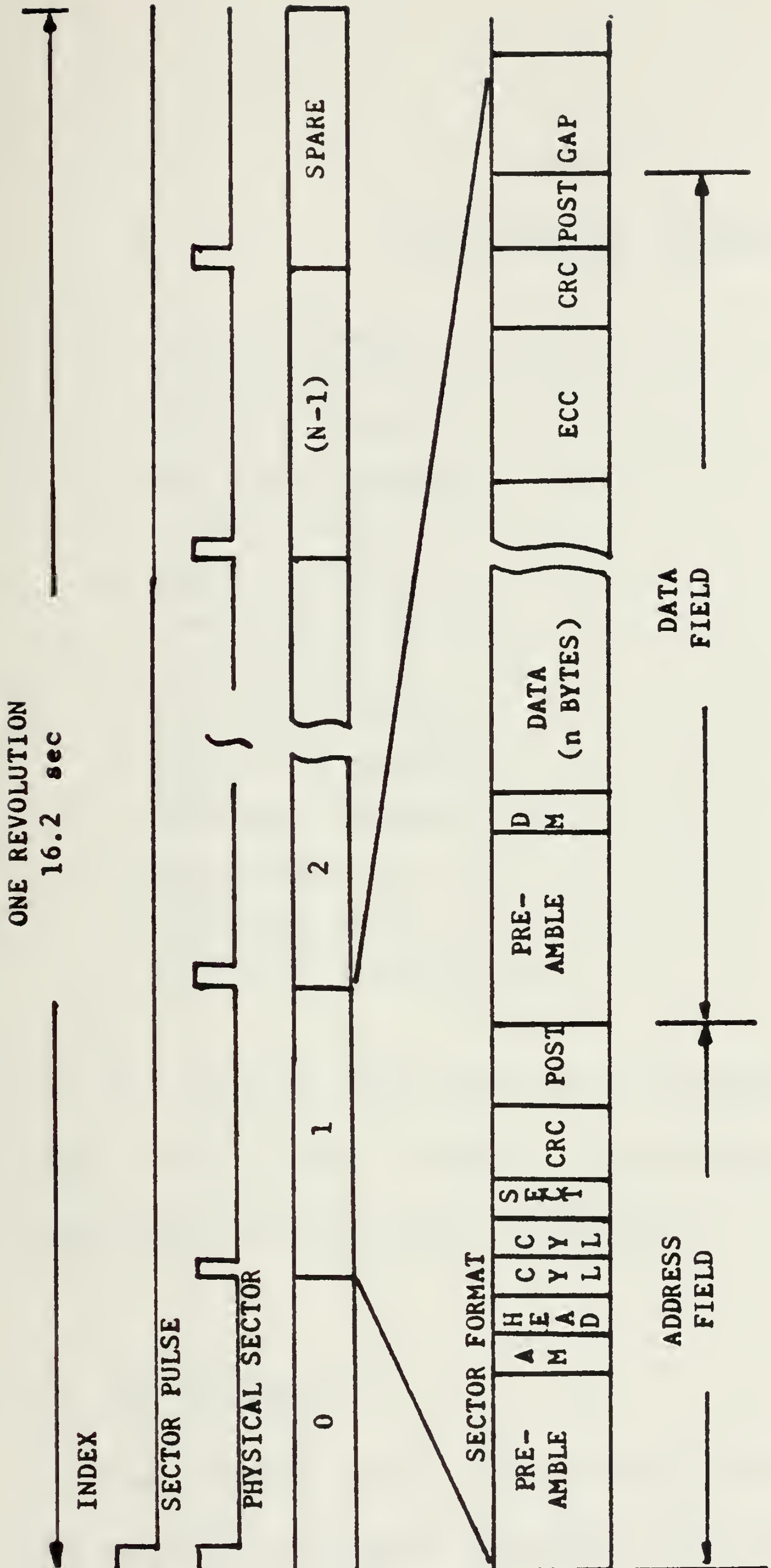
The 1223-1 controller makes use of the track/sector format shown in Figure 2.1. Tracks are divided into a number of sectors which contain a fixed blocklength of user data.

The beginning of each sector is identifed by a sector pulse from the disk drive. Each track contains one spare sector which at the time of initialization can be made to fall over a defictive area of the track.

The sectors are divided into an address field, a data field, and a trailing gap area. Data is recorded most significant bit first where bit 7 is the most significant and bit zero the least significant of each byte. The address field contains a unique track/sector address and associated information. This field is written during initialize commands only. The preamble synchronizes the read circuits. The address mark identifies the beginning of an address field. There are two cylic redundancy checks (CRC) bytes, conputed from the contents of the address mark and bytes 0-3 using the polynomial

$$x^{16} + x^{12} + x^{5} + 1$$

This polynomial catches all single and double errors, all errors with an odd number of zero bits, all burst errors of length 16 or less, 99.997% of 17 bit error burst, and 99.998% of 18 bit and longer bursts. [Ref. 1] Bytes 0 thru 3 contain the head, cylinder, and logical sector addresses. The data field contains user data for transfer to or from



NUM. OF SECTORS (N)	DATA BYTES PER SECTOR (n)	CAPACITY PER TRACK (kB)
66	128	8.4
42	256	10.8
40	268	10.7
24	512	12.3
24	514	12.3
12	1024	12.3

Figure 2.1. Disk Format

TABLE I

Specification Summary

Spindle Speed	3600 rpm
Average Latency	8.33 msec
Track Density	478 tpi
Available Tracks	580
Access Time:	
Track to Track	12 msec
Average(1/3 stroke	42 msec
Full Stroke	85 msec
Sectoring Method	hard
Number of Sectors	24
Encoding Method	EPM(mod.3PM)
Data Density	8623 bpi
Transfer Rate	922 Kbyte/sec(max)
Capacity (formatted)	35.6 Mbytes

the host system. This data field contains the preamble, data mark, data, ECC, CRC, and postamble. The gap provides tolerance for disk speed variation.

C. THE COMMANDS

The command set is divided into three classes:(1) class one which is a nondata transfer, (2) class two command which transfers data from the controller to the host and (3) class

three command that transfers data from the host to the controller. Each data surface of the disk must be prerecorded with the desired format before normal use. The class one command or initialize command is used to initialize the tracks and verify the format. A user utility program is required to initialize then verify each track on the disk with the desired format. This program can be found in Appendix A. The class command byte coding can be seen in Figure 2.2. Bits 0 & 1 define the class, bits 2,3 and 4 are described in Figure 2.2, and bit 5 is not used. Bit 6 is the seek command and bit 7 sets the automatic retry.

The class two read command involves data transfers from the controller to the host. The class two command byte coding can also be seen in Figure 2.2. The four basic commands specified by bits 2 and 3 can be executed in a number of different modes depending on the value of bits 4-7. Table II provides a breakdown of the definitions of each bit of the class 2 command byte.

The write command or class 3 commands pertain to data transfers from the host computer to the disk controller. The class three command byte coding can be seen in Figure 2.2 Bits 4-7 have the same definitions as they did for the read

TABLE II

Read Command Byte

Bits 0,1............	02h. Class 2 code.
Bits 2,3............	Command code:
=1............	Correction. The contents of the sector buffer undergo a correction attempt.
=2............	Read with address check override.
=3............	Normal read.
Bit 4............	Track. Selects logical or physical sector sequencing.
Bit 5............	Selects direct/buffered mode.
Bit 6............	Seek.
Bit 7............	Automatic retry override.

command byte. Bits 0,1 contain a 03H identifying the class command. Bit 2 selects the write or verify command. If bit 2 is equal to one it is the write command. Host data is transferred to the controller and is written onto the disk in the mode specified by bits 3-7. Automatic rewrites occur if bit 3 is a 1. If bit 2 equal 0 implies the verify command. Host data is compared byte-for-byte against data read from the disk. This command is normally used directly

CLASS 1

7	6	5	4	3	2	1	0
RTO	SEEK		CMD			0	1

CMD= 0 DR. STATUS 4 INIT. TRACK
1 SEEK ONLY 5 VERIFY FORMAT
2 READ HEADER 6 INIT. & VERIFY
3 RESTORE 7 FAULT RESET

CLASS 2

7	6	5	4	3	2	1	0
RTO	SEEK	DIRT	TRAK	CMD		1	0

CMD CODE= 0 CORRECT
1 READ WITH DCO
2 READ WITH ACO
3 NORMAL READ

CLASS 3

7	6	5	4	3	2	1	0
RTO	SEEK	DIRT	TRAK	RAW	WRIT	1	1

WRITE= 0 VERIFY
1 WRITE

Figure 2.2. Class Command Byte Coding

after a write command to verify that the data has been correctly recorded. Bit 3 fcr write commands is an automatic read-after-write process performed as each sector is written.

D. PARAMETER AND TERMINATION BYTES

The Micropolis Winchester disk requires six parameter bytes for transmitting address data. The six parameter bytes contain address and control infromation associated with each command. All parameter bytes must be transmitted to disk controller even though some may not be used. A brief description of each parameter byte can be seen in Table III.

The next byte to be presented is the termination status byte. This byte is made available by the disk controller at the end of each command, and contains an error code which identifies an error condition that may have occurred during the command. If zero, the command has been successfully completed; if non-zero the code value indicates the reason for termination.

E. BUS PROTOCOL

1. General Operation

A command on the Micropolis disk is initiated by writing a command byte to the control port, followed by the

TABLE III

Parameter Bytes

Parameter 1..........	Head Address, Unit Address
Parameter 2..........	LSB of Cylinder Address
Parameter 3..........	MSB of Cylinder Address
Parameter 4..........	Starting Sector Address
Parameter 5..........	Sector Spacing Code
Parameter 6..........	Spare Sector Location

six parameter bytes, described in the previous section, and a GO byte to the data port of the controller. The command bytes specifies the type of command, while the parameter bytes contain the associated address information. The GO byte causes the command to be executed and may contain any value. All eight bytes must be transmitted to the controller even though some are not used in certain commands. The use of the GO byte in the command protocol allows the host to ensure the controller has correctly received the command and parameter bytes prior to execution. As each of the command and parameter bytes is received by the controller it is copied into the controller's input buffer and made available to the host. When the GO byte is received, the controller goes busy and proceeds to execute the command. Data transfers between the host/controller/disk take place as required.

2. Host I/O Protocol

Figures 2.3,2.4,2.5 show the I/O bus protocol that must be performed by the host to successfully communicate with the controller. This may be implemented in any combination of hardware/software. The READ data and WRITE data transfer loops given in Figure 2.4 are implemented when the data transfers are performed by a relatively slow host (i.e., in programmed I/O mode by a microprocessor, for example). This protocol applies when transfers are performed in buffered mode.

F. MICROPOLIS INTERFACE REQUIREMENTS

The host interface to the 1223 is made through a 34 pin edge connector. Pinouts and timing requirements are shown in Figures 2.6,2.7. The interface is structured around an 8 bit bidirectional bus and the three control signals WSTR, RSTR, and DATA. Information is output to either a control (command) or data port using write strobe(WSTR), and input from a control(status) or data port using read strobe(RSTR). DATA selects the port in use. These exchanges are controlled by the host making use of handshake flags in the status

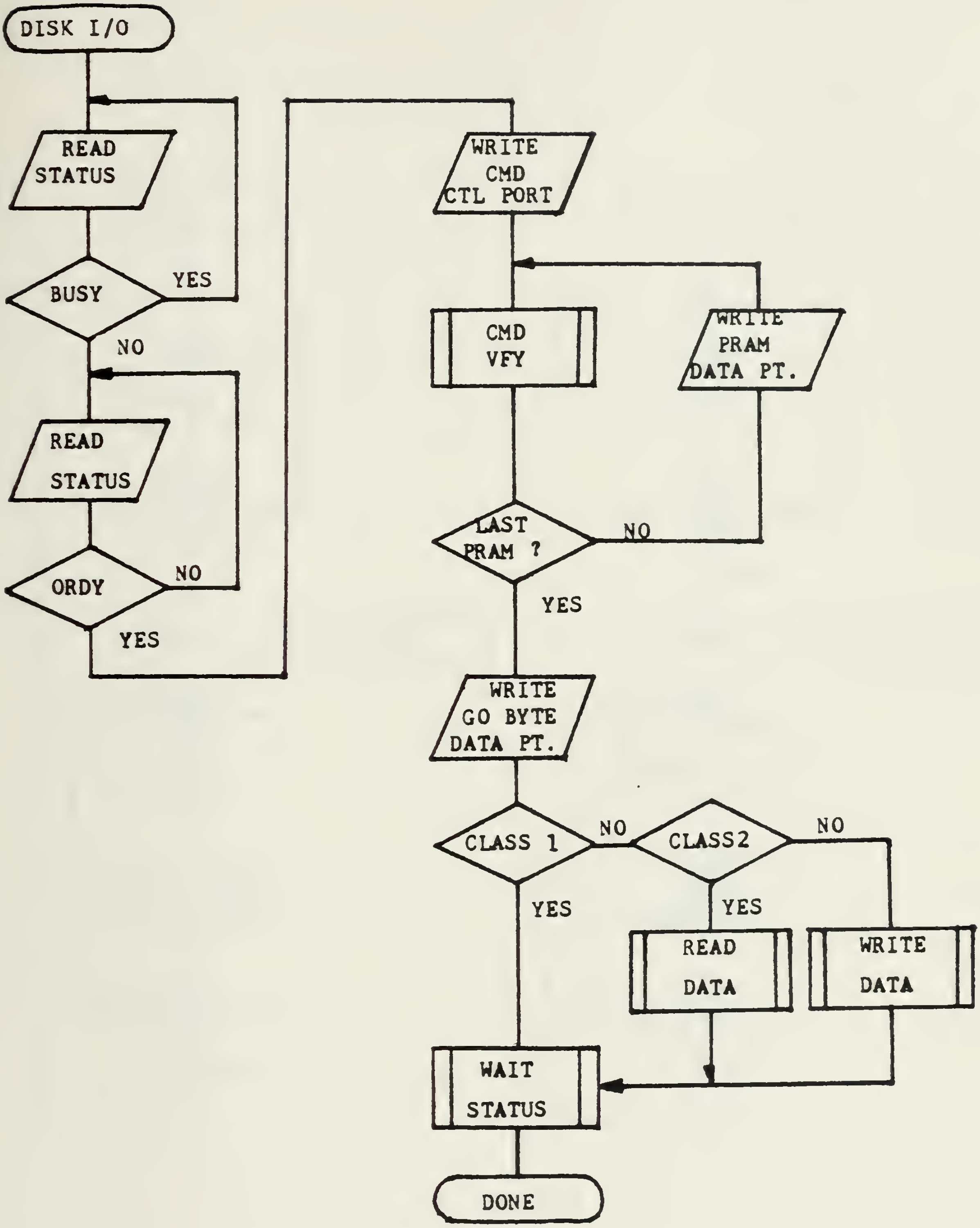


Figure 2.3. Host I/O Protocol

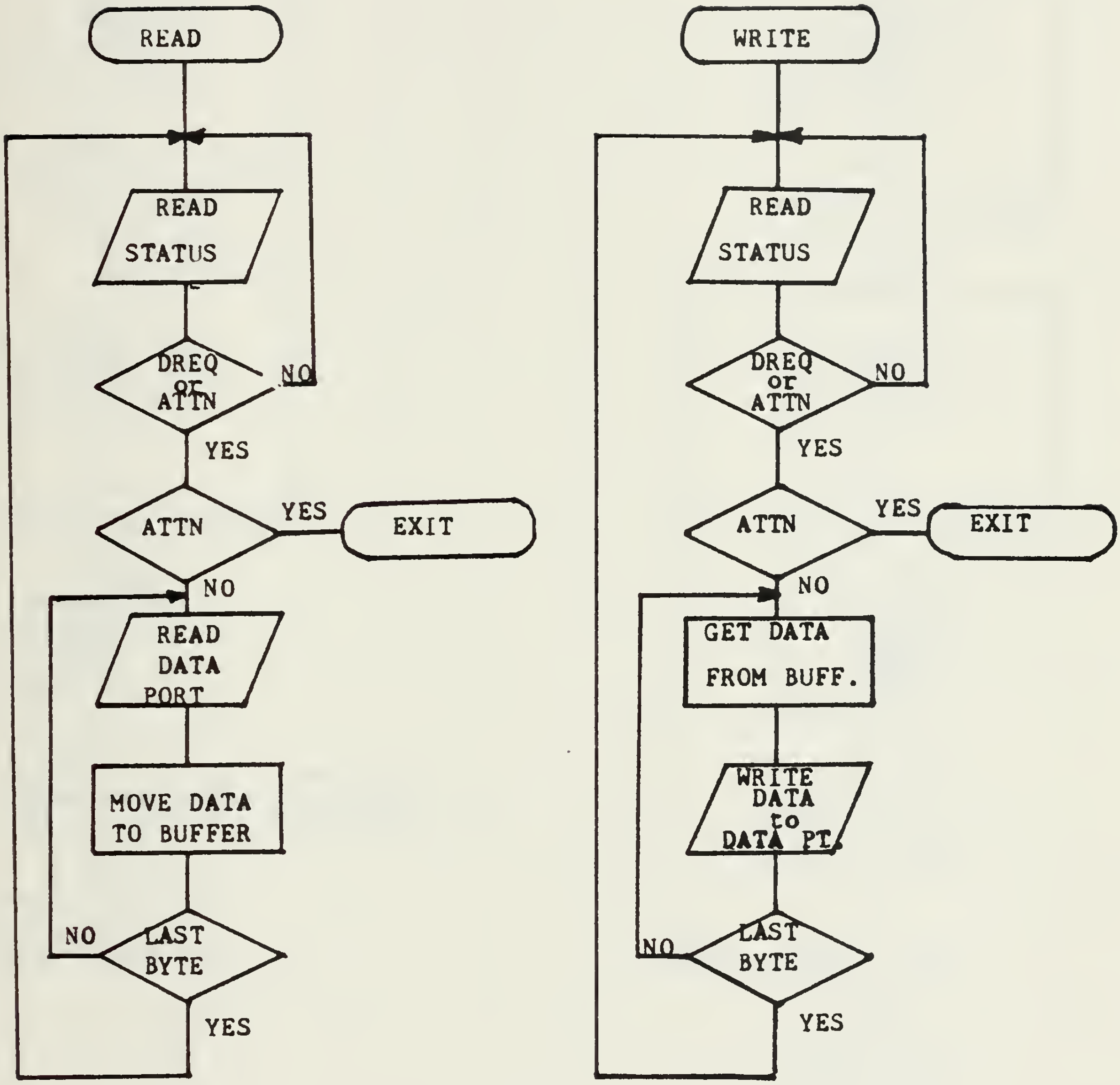


Figure 2.4. Read/Write Protocol

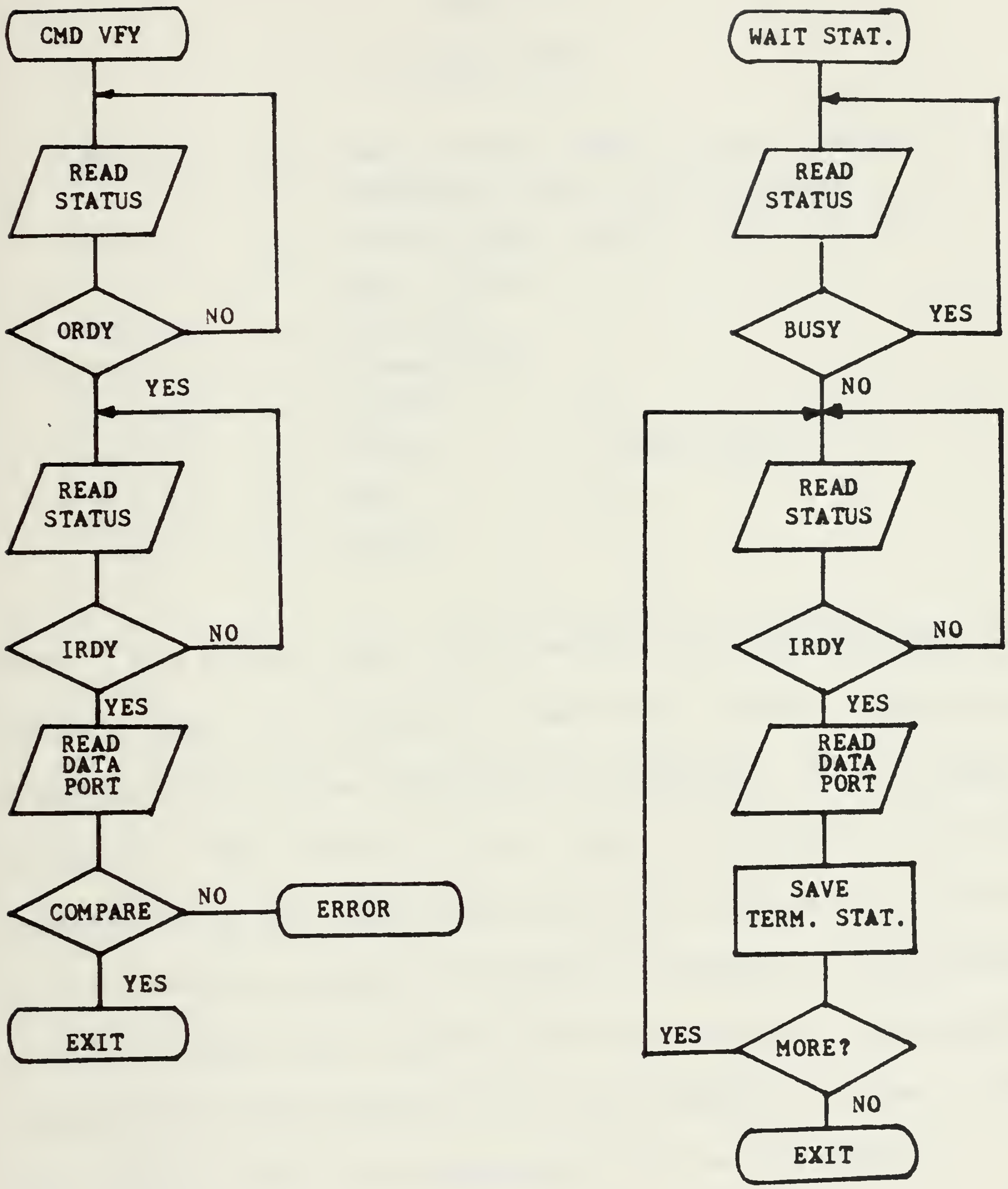


Figure 2.5. Command Verify/Wait Status Protocol

TABLE IV

Status Byte

Bit 0..........	Input ready (IRDY). Input buffer contains a byte for the host.	
Bit 1..........	Output ready (ORDY). Host may output a byte.	
Bit 2..........	Always=1	
Bit 3..........	(Reserved)	
Bit 4..........	CBUSY/	
Bit 5..........	DREQ	See Table V.
Bit 6..........	OUT	
Bit 7..........	ATTN	

byte. The status byte is accessed by reading from the control port. It contains controller status information which coordinates the exchange of information with the host. The status byte coding can be seen in Figure 2.8 and the description of the individual bits are summarized in Table IV The interface signals described in Figure 2.6 are defined in Table V With the knowledge of the interface requirements presented in this section it is now necessary to look at the host computer and it's requirements for an interface.

TABLE V

Interface Signals

SEL..........Selects the addressed disk controller

ENABLE.......Normally held true. Used for programmed reset.

BUS0-7.......Bidirectional tristate 8 line bus.

WSTR.........Write strobe.

RSTR.........Read strobe.

DATA.........Selects the control or data port.

CBUSY/.......Controller busy. Cleared when command issued, set when command is terminated.

ATTN.........Attention. Set true at the end of each command when CBUSY/ changes.

DREQ.........Data request. This flag requests the transfer of each byte of user data to/from the controller.

OUT..........Specifies the direction of data transfer.

J101 CONNECTOR PIN SIG	GND	NAME	DESCRIPTION	SOURCE
2	1	BUS7/	(most significant)	Host/Controller
4	3	BUS6/		
6	5	BUS5/	Bi-	
8	7	BUS4/	Directional Data	
10	9	BUS3/	Bus	
12	11	BUS2/		
14	13	BUS1/		
16		BUS0/	(least significant)	Host/Controller
15		----	(Reserved)	----
18	17	ATTN/	Attention	Controller
20	19	DATA/	DATA/CONTROL SELECT	HOST
22	21	RSTR/	READ STROBE	HOST
24	23	WSTR/	WRITE STROBE	HOST
26	25	ENABLE/	CONTROLLER ENABLE	HOST
28	27	SEL/	CONTROLLER SELECT	HOST
30	29	CBUSY	CONTROLLER BUSY	CONTROLLER
32	31	DREQ/	DATA REQUEST	CONTROLLER
34	33	OUT/	DIRECTION of DATA TRANSFER	CONTROLLER

Figure 2.6. Host Interface Pinout

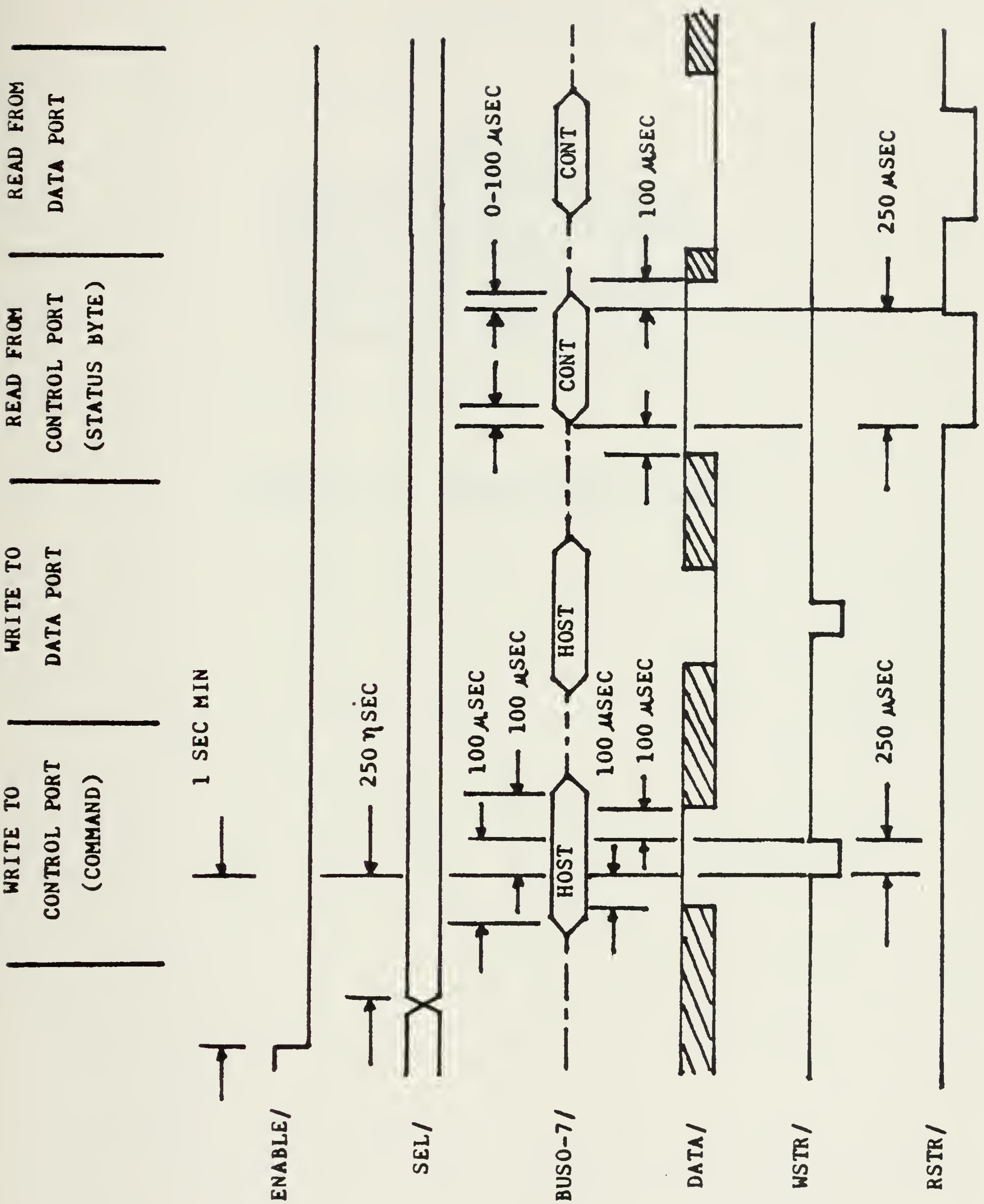


Figure 2.7. Host Interface Bus Timing

STATUS BYTE

7	6	5	4	3	2	1	0
ATTN	OUT	DREQ	CBUSY		1	ORDY	IRDY

Figure 2.8. Status Byte Coding

III. THE INTEL 80/20 SBC

A. SBC CHARACTERISTICS

For the purpcse of this thesis the author feels a brief general description of the Intel 80/20-4 is in order. This is followed by a more in depth presentation of its' I/O capabilities which will prove more useful in the actual interface design that follows.

The SBC 80/20-4 is a member of Intel's line of self-contained computers based on the powerful 8-bit n-channel MOS 8080A CPU. The SBC 80/20-4 is a complete computer system on a single 6.75 by 12 inch printed circuit board. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface,interval timer, bus control logic and drivers all reside on the board. The 8080A has a 16 bit program counter which allows direct addressing of up to 64K bytes of memory. An external stack, located within any portion of memory, may be used as a last in/first out stack to store and retrieve the contents of the program counter, flags, accumulator and all of the six general purpose registers. A sixteen bit stack pointer controls the addressing of this external stack. Sixteen line address and

eight line bidirectional data buses are used to facilitate easy interface to memory and I/O.

A programmable serial communications interface using Intel's 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART) is contained on the board. The USART can be programmed by the system's software to provide virtually any serial data transmission technique presently in use. The 8251 provides full duplex, double buffered transmission and receive capability.

The SBC contains 48 programmable parallel I/O lines implemented using two Intel 8255 Programmable Peripheral Interface (PPI) devices. The software is used to configure the I/O lines in combinations of unidirectional input/output, and bidirectional ports. Therefore, the I/O interface may be customized to meet specified peripheral requirements. In order to take advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators.

B. SBC INTERFACE REQUIREMENTS

1. The 8255 PPI Operational Summary

For the interface considerations presented in the preceding sections the disk controller dictates an 8 bit

bidirectional bus for data transfer. With this constraint in mind only the 8255 PPI need be discussed in detail.

The parallel I/O interface logic on the SBC 80/20-4 provides 48 signial lines for the transfer and control of data to or from the peripheral devices. Sixteen lines have a bidirectional driver and termination networks perminantly installed. The remaining thirty-two lines are uncommitted. Sockets are provided for the installation of active or passive driver/termination networks. The optional drivers and terminators are installed in groups of four by insertion into the 14 pin sockets. A basic block diagram of a single 8255 PPI can be seen in Figure 3.1. The two 8255 devices allow for a wide varity of I/O configurations.

The 8255 contains three 8 bit ports (A,B, and C). All can be configured in a wide varity of functional characteristics as described in Table VI. The 8080 CPU dictates the operating characteristics of the ports by outputting control words to the 8255.

There are three basic modes of operation that can be selected by the system software. Mode zero is the basic input/output mode. Mode one is concerned with a strobed input/output while mode two is the bidirectional bus mode.

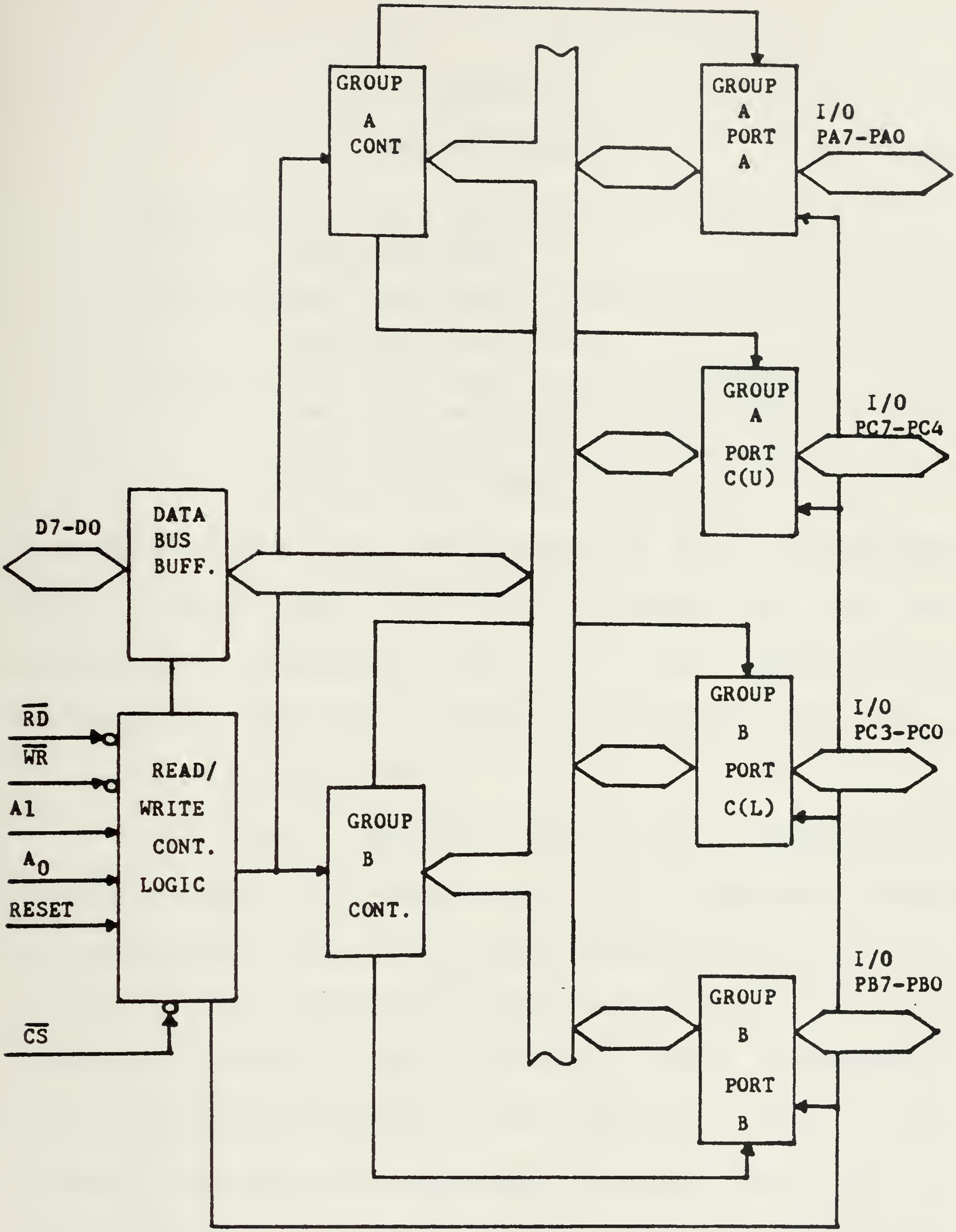


Figure 3.1. 8255 Block Diagram

TABLE VI

Port Definitions

Port A....	One 8 bit data output or input latched buffer.
Port B....	One 8 bit data I/O latch buffer and data input buffer.
Port C....	n25 8 bit data output latch buffer one 8 bit data input buffer. This port can be divided into two 4 bit ports under mode 2 operations.

A summary of the mode deffinitions and port configuration can be seen in Table VII. Due to the fact that the disk controller's requirement for an 8 bit bidirectional datalines it will only be necessary to discuss the mode 2 operation of the 8255 here.

The mode 2 bidirectional bus I/O configuration provides a means for communicating with a peripheral device or structure on a single 8 bit bus for both transmitting and receiving data. Handshaking signals as seen in Table VII are proviged to maintain proper bus flow. Interrupt generation and enable/disable functions are also available. It is apparent from Table VII that mode 2 is only used in port A. Port C provides the five bit control port while port B can

TABLE VII

Mode Definition Summary

	MODE 0		MODE 1		MODE 2
	IN	CUT	IN	OUT	GROUP A ONLY
PA0	IN	OUT	IN	OUT	BIDIRECTIONAL
PA2	IN	CUT	IN	OUT	BIDIRECTIONAL
PA3	IN	OUT	IN	OUT	BIDIRECTIONAL
PA4	IN	CUT	IN	OUT	BIDIRECTIONAL
PA5	IN	OUT	IN	OUT	BIDIRECTIONAL
PA6	IN	OUT	IN	OUT	BIDIRECTIONAL
PA7	IN	CUT	IN	CUT	BIDIRECTIONAL
PB0	IN	CUT	IN	OUT	------------
PB1	IN	OUT	IN	OUT	------------
PB2	IN	OUT	IN	CUT	------------
PB3	IN	CUT	IN	OUT	------------
PB4	IN	OUT	IN	OUT	------------
PB5	IN	OUT	IN	OUT	------------
PB6	IN	OUT	IN	OUT	------------
PB7	IN	CUT	IN	OUT	------------
PC0	IN	OUT	INTR(B)	INTR(B)	I/O
PC1	IN	OUT	IBF(B)	OBF(B)	I/O
PC2	IN	CUT	STB(B)	ACK(B)	I/O
PC3	IN	OUT	INTR(A)	INTR(A)	INTR(A)
PC4	IN	OUT	STB(A)	I/O	STB(A)
PC5	IN	OUT	IBF(A)	I/O	IBF(A)
PC6	IN	CUT	I/O	ACK(A)	ACK(A)
PC7	IN	OUT	I/O	OBF(A)	OBF(A)

be used in mode 0 or 1. It should also be noted that both the inputs and outputs are latched. A high on the INTR (Interrupt Request) output can be used to interrupt the CPU for both input or output operations. The OBF (Output Buffer Full) output will go low to indicate that the CPU has written data to port A. A low on the ACK (Acknowledge) input enables the tristate output buffer of port A to send out the data. A low on the STB (Strobed Input) indicates that data has been loaded into the input latch while IBF (Input Buffer Full) output indicates that data has been loaded into the input latch. It might be pointed out at this time that all or none of the handshaking signals just presented can be used for the 80/20 to function properly.

C. THE INTELLEC MDS SYSTEM

The Intellec MDS was used in this thesis as the design center for the 80/20 SBC. The Intellec is a complete microcomputer design system that provides total support through the entire production design cycle. The MDS is also modular, which allows custom tailoring of systems.

The standard Intellec MDS system has four main components: (1) central processor, (2) frount panel control unit (3) a monitor module and (4) 16K RAM. The central processor

of the MDS system is an Intel 8080 with the same capabilities as the SBC 80/20 discussed earlier. The processor was used to develope the software and provide a means of loading the interface programs into the 4K RAM of the SBC. Memory and I/O interface logic is also provided on the CPU module. The module drives a three state, 16 line address bus, which communicates with the external memory and I/O device decoding logic. A bidirectional, 8 line data bus provides the means for the actual data transfers. The CPU module can address up to 65,536 bytes of memory. The 16K RAM module provides the Intellec MDS system with 16,384K by 8 bit words of dynamic random access memory. There are four RAM modules used in the MDS utilized in this thesis, for a total of 64K RAM. The monitor module of the resident CPU was not used for the interface developement but the monitor that resided on the SBC was used, therefore a brief description is in order. The monitor module enables the MDS system to have firmware storage for the monitor program and I/O interfaces with peripheral devices such as teletype, CRT, line printers, or paper tape readers. The monitor module can include 2048 by 8 bit words of ROM for storage of the system monitor program. The monitor program used on the SBC is

identical to the DDT(Dynamic Debugging Tool) program associated with Digital Research's CPM operating system. The monitor module in conjunction with a CRT for instance can be used to control the transfer of data, control, and status information between the SBC and it's associated I/O device. Here again it must be pointed out that the monitor of the MDS system was not used but the monitor on the SBC, which is identical, was utilized. The frount panal control module drives the INTERRUPT, RUN and HALT switches. This module served to provide a means of interrupting the execution of a program to allow the user to monitor the progress of the program or check the contents of the registers.

IV. THE INTERFACE DESIGN

A. HARDWARE

In the preceding chapters the characteristics and interface requirements for the hardware involved in this thesis was presented. In this chapter the author describes the actual interface used and the software developed to successfully communicate with the disk.

1. The Micropolis1223-1

The first consideration for building the hardware interface for the Micropolis disk was to determine which of the provided handshaking lines would be required to operate the system in the buffered mode. Using the descriptions provided in Figure 2.6 and Table V it is apparent that the signals which source is the host would be necessary for the disk controller to function properly, but those which source is the controller could be implimented in the software. Inotherwords, the signal lines ATTN, CBUSY, DREQ, and OUT are also flags in the status byte. These lines were provided for the flexibility of operating the disk in a DMA environment. Inaddition, the only other lines required were the SEL and ENABLE lines. Using the definition of SEL from Table V this line was connected permanently to a +5V source

because there was no other disk controller in the system. The ENABLE was used as a programmed reset at the beginning of each execution of a read or write command.

2. Intel 80/20 SBC

It was established in Chapter III that the SBC's PPI would be required to operate in mode 2 to satisfy the bi directional needs of the Winchester disk. Looking at the signals available at port C of the PPI in the mode 2 column of Figure 3.1 it is conveniant to utilize the three I/O lines PC0-PC2 to accommadate the WSTR, RSTR, and DATA lines. This is an obvious decission for two reasons. First of all port C in mode 2 can be divided into 2 four bit ports leaving port B available for mode 0 or 1 operations for the remaining control lines. Secondly, since PC0-PC3 belong to port C(L) it has an available socket (A4) which is ideal for the terminator network that is required by the disk; more on this later under electrical considerations. The remaining handshaking signals were specifically designed for a DMA mode. Since the PPI was programmed for bidirectional bus transfer and operating in a buffered mode the output signals OBF, IBF were not needed. Likewise, since the SBC was the only device requiring access to the Winchester disk the INTR

line was not employed. The SBC does require 2 inputs for proper operation those being ACK, and STB. Using the description of RSTR in Chapter II it is the ACK signal required by the SBC. This implies that all that is necessary for this particular handshake is a feedback of RSTR to the SBC on every read command. The ENABLE line mentioned in the preceding section was designed as a reset taking it's inputs from port B at PB0 and passing it through a termination device before connecting it to the disk controller. A basic block diagram of the data and handshaking lines can be seen in Figure 4.1 and the interface pinouts are described in Figure 4.2.

B. ELECTRICAL CONSIDERATIONS

The 1223 requires the same D.C. supply voltages as an industry standard 8 inch flexible disk drive. The Winchester disk was mounted in a dual floppy disk frame. This was done with only a slight modification to the mounting brackets. The interfacing of the handshaking signals required buffer/driver gates with an open collector output. The DM7438 is a quad dual input NAND gate with the desired open collector feature which made it ideal for the control lines.

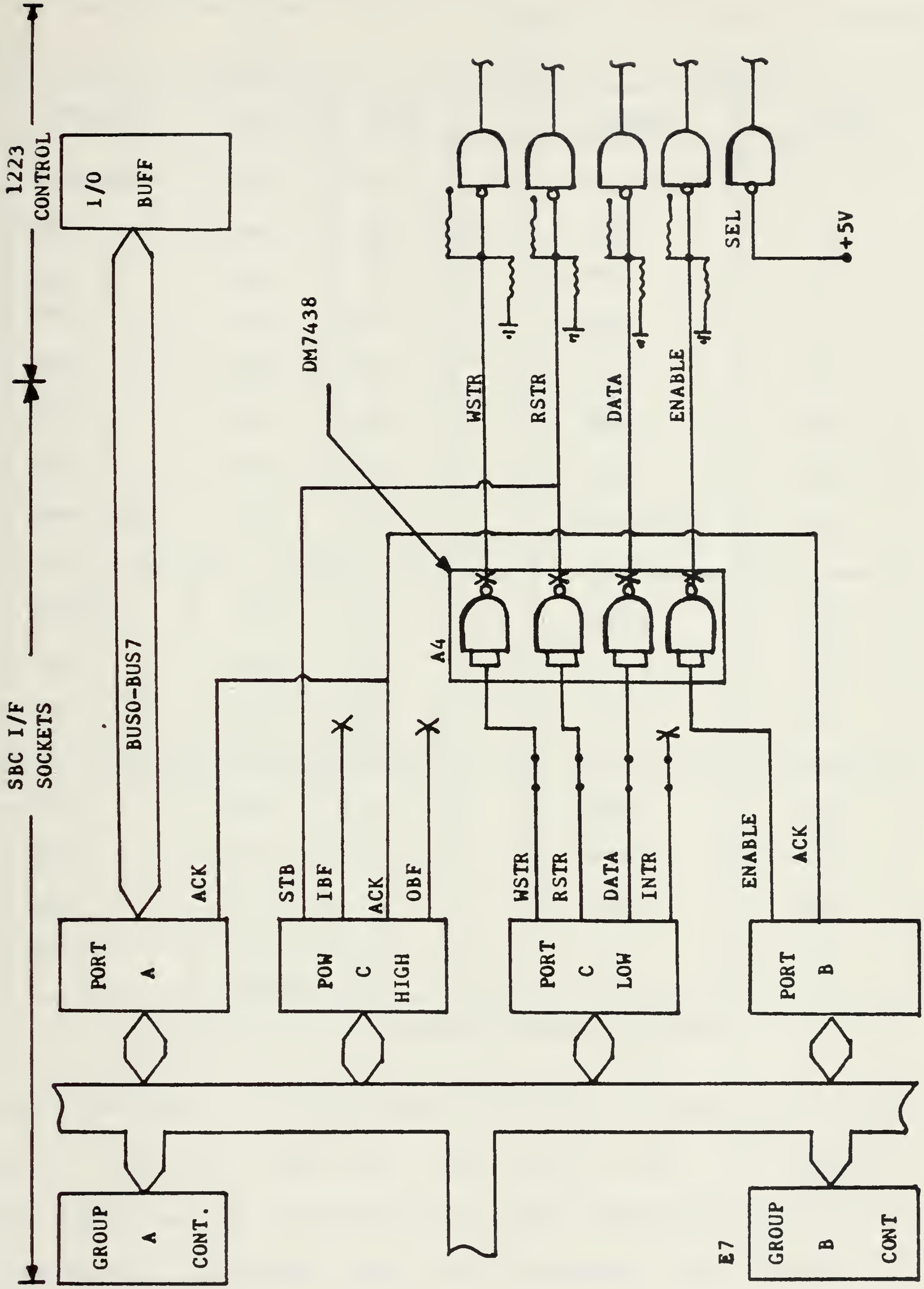


Figure 4.1. Interface Block Diagram

HOST				DISK	
FUNCTION	PORT	PIN#		FUNCTION	PIN#
DATA (MSB)	PA7	34		DATA (MSB)	2
DATA	PA6	36		DATA	4
DATA	PA5	38		DATA	6
DATA	PA4	40		DATA	8
DATA	PA3	42		DATA	10
DATA	PA2	44		DATA	12
DATA	PA1	46		DATA	14
DATA (LSB)	PA0	48		DATA (LSB)	16
ENABLE	PB0	16		ENABLE	26
SEL	PB1	14		SEL	28
ACK	PB2 (F/B TO PC6)	12		----	--
WSTR	PC0	24		WSTR	24
RSTR	PC1	22		RSTR	22
DATA	PC2	20		DATA	20
STB	PC4 (F/B to PC4)	22		----	---

Figure 4.2. System Interface Pinout

The electrical hcst interface used can be seen in Figure 4.3. The SBC 80/20 had the 8226 four bit parallel bidirectional bus drivers with their associated 1K pullup resistors installed. Also each line out of the SBC is

inverted as is the input of the disk controller eliminating any need for inverters in the data lines. The actual connection was achieved through a 34 pin flat cable attached to the disk controller edge connector J101.

C. SOFTWARE

1. Initialization and Verification

Each data surface of the disk must be prerecorded with the desired format before normal use. Three initialize commands are provided for this purpose. These commands can be reviewed in section C of Chapter II. This program can be seen in Appendix A. The initialize and verify program (INTVFY.ASM) combines the two commands INITIALIZE TRACK and VERIFY FORMAT into one command (19H). The flow diagram of Figure 2.3 was used to implement this user utility program, with some slight modifications. The decision loops at the bottom of the flow diagram for read and write commands were eliminated to prevent an accidential INTVFY being executed which would destroy any user data on the disk. This is some what redundant due to the fact that command echoing is used in the protocol tc prevent just such errors. The intialize and verify routine does not envolve any data transfers between the disk controller and the host. The INTVFY program

is designed to format one entire platter side by holding all parameter bytes fixed except the cylinder address bytes. Once the single disk side is completed (approx. 18 sec) the user can change the head address parameter byte using the monitor on the SBC with the substitute (S) command and then restarting the program. The entire disk drive can be reformated in 5 minutes using this technique.

The read and write program is a variant from the INTVFY.ASM program in that it is comprised of a series of subroutines whereas INTVFY.ASM is a single string without any branching. Subroutines were written for the most frequently used modules such as IRDY, ORDY, STATUS, and CBUSY. This proved to be slower than ideal execution time due to the number of PUSH and POP commands that are enherant with subroutines. The read or write program (READRITE.ASM) can be seen in Appendix B.

D. SOFTWARE TIMING

The execution of a controller command consists of three phases: initiation, execution, and termination. In the intiation phase, the controller decides the command specified by the host, verifies the validity of parameters and performs housekeeping functions necessary to execute the

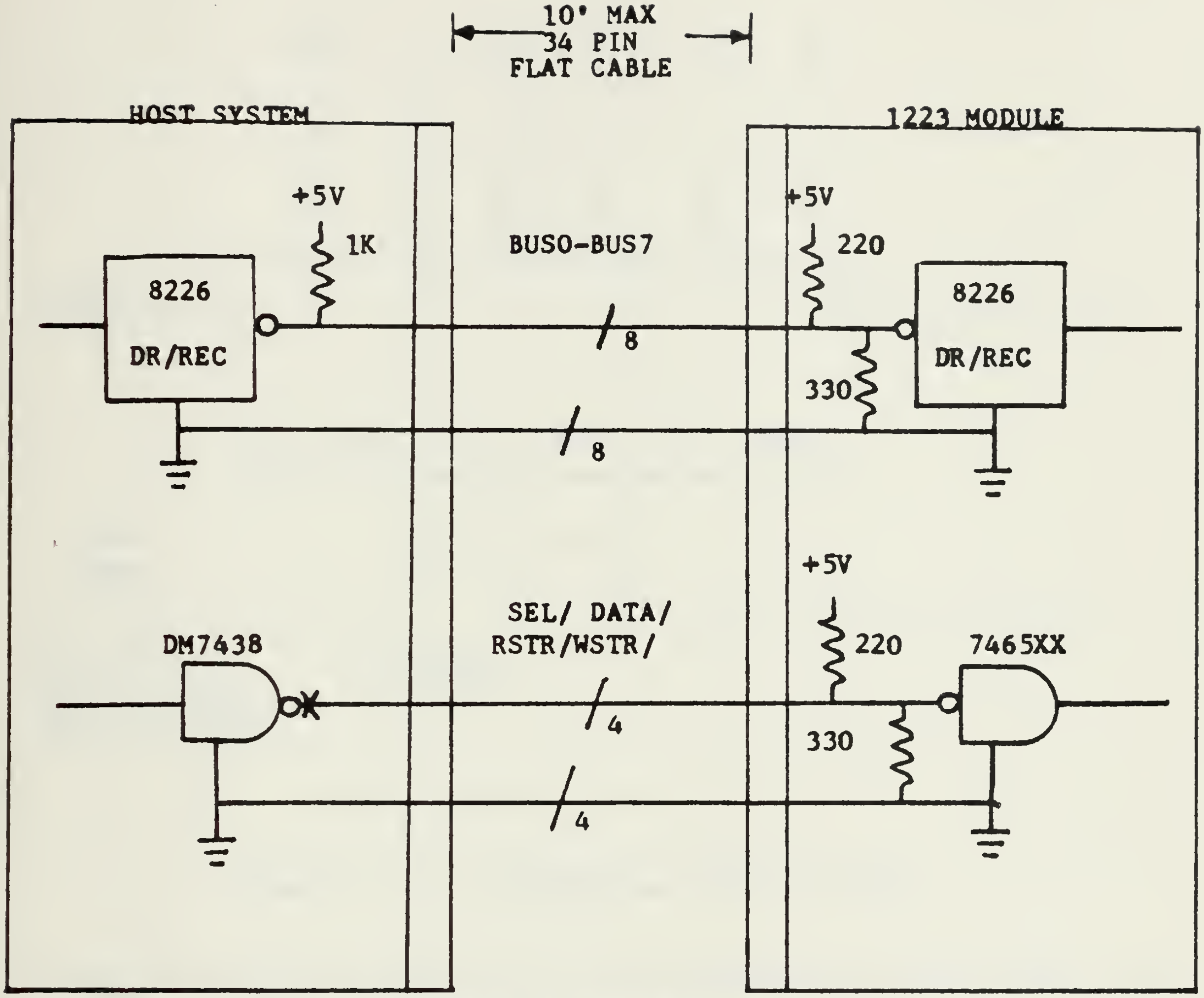
command. In the execution phase, the requested functions are performed. In the termination phase, the controller performs post execution housekeeping and determines the termination status for the command.

An example of the software timing, as required by the delays specified in Figure 2.7, can be seen in the ORDY subroutine in Appendix B. The majority of the software timing involved an extensive use of the IN and OUT commands since the program is basically concerned with I/O data manipulation. By outputting a 02 hex to the SBC control port E6, in this example, the RSTR pulse is turned on at the control port (port C(L)). This followed with a 00 hex to the same port turns off the RSTR. The user is also reminded that anytime a read strobe control word is being pulsed that this is also pulsing the STB control line of the SBC. This latches in the the disk controller status byte which is then moved into the accumulator to mask the ORDY bit (bit 1 of status byte).

An example of writing a command byte to the command port can be seen in Appendix A. Using the timing diagram Figure 2.7 to write a command byte to the disk controller command port WSTR needs to be pulsed and the ACK to the SBC's 8226

must be turned on before pulsing and then off after strobing is completed. The pulsing sequence can be seen on line 70 of Appendix A. Here the command INTVFY is being sent to the disk controller's command port. The command is first latched into the SBC's data port (E4). The ACK line is next turned on at the SBC control port E5 followed by the strobing of WSTR. The sequence is completed by restoring the 8 DATA lines to the input mode by turning ACK off.

The next example is the writing of a parameter byte to the disk controller data port. This can best be demonstrated by looking at the PRAM1 module of the INTVFY program in Appendix A. For this particular case the parameter byte to be sent out contains all zeros. By using Table III this implies that the head and unit address is zero. The pulsing here is the same as it was in the preceding example except that the data control line of E6 is pulsed prior to the WSTR line pulsing and turned off upon completion of the WSTR pulse going low. This technique complies with the timing delays in Figure 2.7. The measured pulse delays are in agreement with Figure 4.4 which is the calculate values of the delays.



1. All signal lines are low true at the interface connector and high true into drivers and out of receivers.

2. Interface signal levels are low= 0-0.4V@25mA.
high= 2.5-5.0V@0mA

3. Host provides 1K pullups on Bus 0-Bus7.

4. 220/330 ohm terminators are installed in 1223 module.

Figure 4.3. Electrical Interface

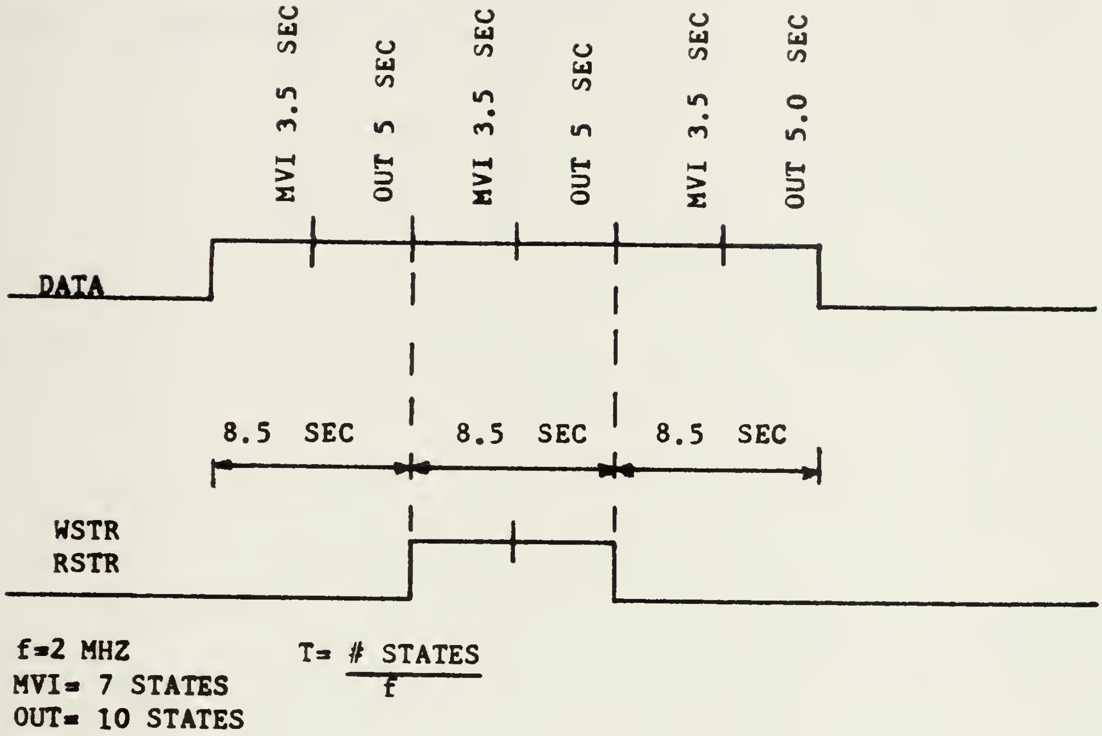


Figure 4.4. Calculated Pulse Timing

When the disk controller has received and verified the command byte, six parameter bytes, and the GO byte the disk controller goes low and executes the command. The disk then transfers to or from the host 512 bytes of user data. This data is then stored in the buffer titled TABLE1 in the program in Appendix B. Upon completion of the data transfer the disk controller finishes up the termination phase by

issuing the termination status byte to the host computer. A breakdown of the termination status byte error codes can be seen in Reference 2.

V. CONCLUSIONS AND RECOMMENDATIONS

From Chapter IV it would appear that the interface design was a clear and strightforward process. However, the author encountered several inconsistancies that made the job not so candid. The facts that the author was unfamiliar with the hardware and had very limited exposure to assembly language programming techniques compounded the task.

A. INTERFACE DIFFICULTIES

One of the first difficulties encountered in the design phase of the interface was how to best utilize the large number of handshaking lines and to determine which ones would not be necessary for operations in the buffered mode. Too few of the lines had the same definitions or functions compounding the problem. The documentation provided for the SBC 80/20 was confusing caused mainly by the number of options, modes, and port definitions that are available. The documentation pertaining to the Micropolis disk was the author's biggest stumbling block. The manual refered to "track oriented" commands and "noraml commands". It took a number of hours of reading and rereading the manual in

conjunction with phone calls to the manufacturer to resolve the difference. It was at the last possible minute that the author was able to ascertain, from the manufacturer, a statement that the AUXILLARY STATUS bytes, which contain detailed drive and controller status information, was incorrect in the manual and that two revisions had been made to the text.

The MDS system provided additional hardware problems. At the outset of the project the SBC was being used on the double density MDS system. This being the only double density system at NPS a waiting list to use the system was necessary. Once a dedicated MDS system was assigned to the project the problem was traded for another. The second system is a single density version MDS which possessed the frustrating cronic habit of crashing the O/S and the directory once a week not to mention burning out the power supply. The author lost 8 hours a week just recovering from these failures and updating backup disks.

B. RECOMMENDATIONS

One of the recommendations for future hard disk operations in the AEGIS modeling system would be to modify the system presented here to allow the disk to operate in

the DMA mode. This would require some hardware and software alterations to the present system.

The hardware changes would include making use of handshaking lines provided as four output lines of the disk controller namely: ATTN, CBUSY, DREQ, and OUT. These four lines could be connected to the A port of the second PPI on the SBC. Using port A would eliminate any need for adding drivers or terminator networks because they are already installed. Then by polling this port it would eliminate the need to read in the status byte after every transfer of a byte to check for flags. Two additional inputs would be required on the host computer side for OBF and IBF to prevent an overrun of data during transfers.

The read and write command software would need only a slight modification. The READ and WRITE flow diagrams presented in Figure 2.4 would be modified to conform to the flow diagrams in Figure 5.1. These read data and write data transfer loops would provide a general transfer protocol which is insensitive to sector length, number of sectors being transferred, and the speed of the host interface. In the direct mode, the host interface must provide for response to all data requests at disk speed.

Before the Micropolis disk can be fully implemented in the AEGIS modeling system it will require a Customized Basic Disk Operating System (CBIOS). The author had originally intended to include a CBIOS as an appendix to this thesis but was unable to do sc because of the number of hardware failures of the MDS system.

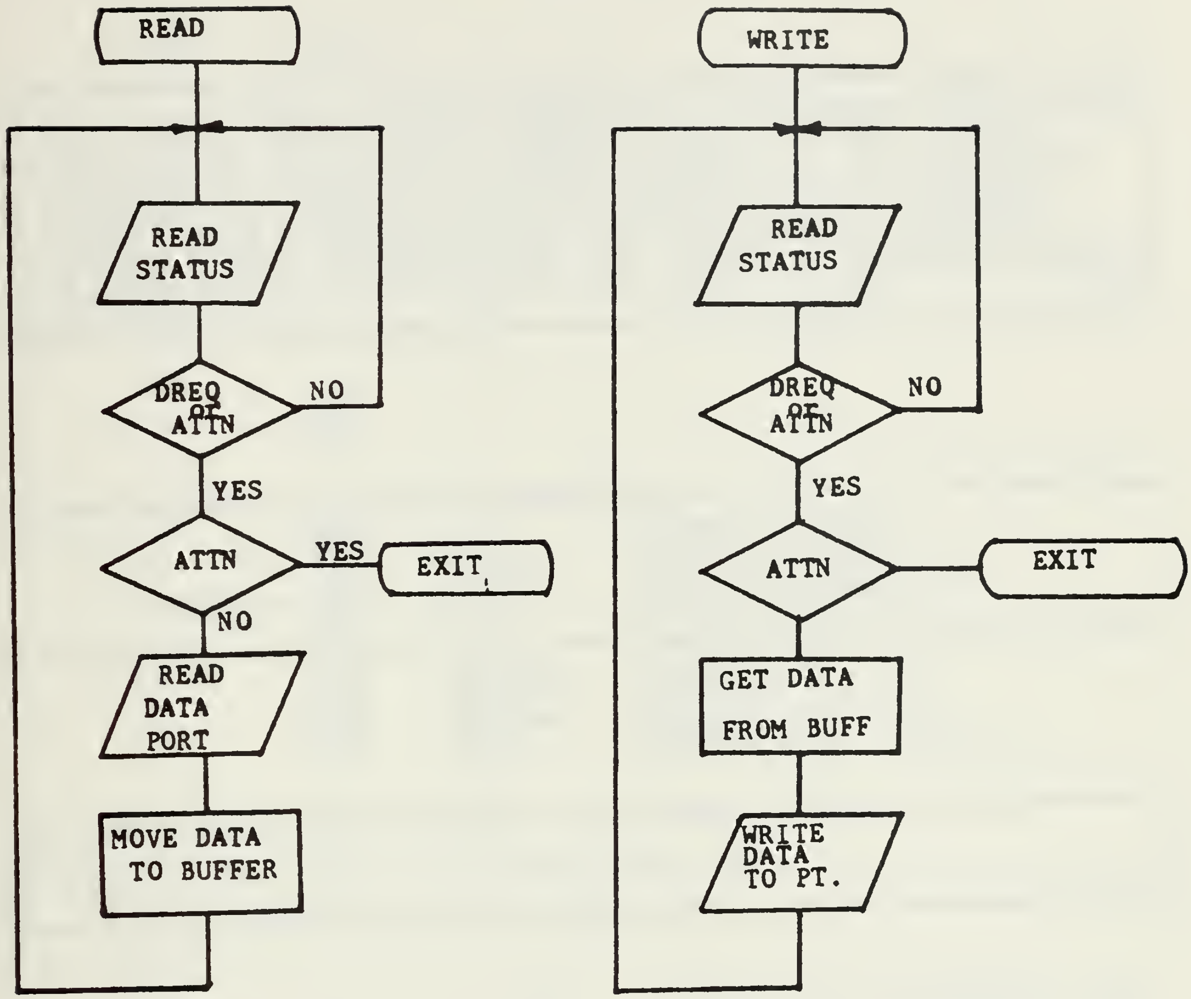


Figure 5.1. Alternative Data Transfer Protocol

APPENDIX A

```
        ORG     3000H

;*****************************************************************
;* THIS IS A USER UTILITY PROGRAM TO INITIALIZE THEN VERIFY*
;* EACH TRACK ON THE 1223 DISK WITH THE DESIRED FORMAT.     *
;* INITIALIZE TRACK WRITES ENTIRE LENGTH OF CURRENT TRACK   *
;* USING HEAD, CYLINDER, SECTOR SEQUENCING, AND SPARING     *
;* INFORMATION CONTAINED IN THE ACCOMPANYING PARAMETER      *
;* BYTES. DATA FIELDS CONTAIN 51H IN ALL DATA LOCATIONS.    *
;* VERIFY FORMAT VERIFIES THAT THE TRACK IS CORRECTLY INIT *
;* IALIZED READS ENTIRE TRACK AND COMPARES AGAINST ORIGINAL*
;* PATTERN......                                             *
;*****************************************************************
;
;
;
                NOP
                NOP
;
;
;*****************************************************************
                SUB     A           ;CLEAR ACCUM...
                STA     PRAM2       ;ZERO OUT PRAM2
                STA     PRAM3       ;ZERO OUT PRAM3
        INTVFY: ADI     6           ;LOAD COUNTER INTO
                MOV     B,A         ;B REGISTER.
;*****************************************************************
                MVI     A,0C0H      ;PROGRAM 8255 TO
                OUT     0E7H        ;MODE 2.
                MVI     A,004H      ;INITIALIZE ACK/
                OUT     0E5H        ;OUTPUT ACK/ TO PT. B
;
;
;*****************************************************************
;*                                                               *
;* READ STATUS BYTE. IS CONTROLLER BUSY.....?                    *
;*                                                               *
;*****************************************************************
;
;
;
     CBUSY1:    MVI     A,002H      ;RSTR CMD TO CONTROL PORT
                OUT     0E6H        ;
                MVI     A,000H      ;RSTR PULSE OFF
                OUT     0E6H
                IN      0E4H        ;READ STATUS WORD
                ANI     010H        ;IS CBUSY TRUE
                CPI     010H        ;OR FALSE
                JNZ     CBUSY1      ;CONTROLLER BUSY GO BACK..
;
;
;*****************************************************************
;*                                                               *
;* READ STATUS BYTE TO SEE IF OUTPUT BUFFER IS FULL..            *
;*                                                               *
;*****************************************************************
;
;
      ORDY1:    MVI     A,002H      ;RSTR CMD TO
                OUT     0E6H        ;CONTROL PORT.
                MVI     A,000H      ;PULSE RSTR ON
                OUT     0E6H        ;THEN OFF.
                IN      0E4H        ;READ STATUS BYTE
                ANI     002H        ;MASK STATUS BYTE FOR ORDY.
                CPI     002H
                JNZ     ORDY1       ;RETURN TO RSTR IF NO ORDY.
```

```
;
;*************************************************************
;*                                                           *
;*  LOAD COMMAND BYTE  1 INTO CONTROLLER FOR INITIALIZATION *
;*  AND VERFICATICN OF DISK...                               *
;*                                                           *
;*************************************************************
;
                MVI     A,019H      ;LOAD CMD 1
                CUT     0E4H        ;PUT CMD 1 TO OUTPUT PT.
                MVI     A,000H      ;TURN ACK/ ON
                OUT     0E5H         PORT B
                MVI     A,001H      ;WSTR CONT. WORD TO ACCUM.
                OUT     0E6H        ;PULSE WSTR ON
                MVI     A,000H      ;PULSE WSTR OFF
                OUT     0E6H
                MVI     A,004H      ;RESTORE ACK/ TO
                OUT     0E5H        ;B PORT
;
;*************************************************************
;*                                                           *
;*  CCMMENCE COMMAND VERIFY CF COMMAND BYTE BY READING      *
;*  STATUS BYTE...                                           *
;*                                                           *
;*************************************************************
;
    ORDY2:      MVI     A,002H      ;RSTR CMD TO
                OUT     0E6H        ;CMD PORT
                MVI     A,000H      ;ON AND
                CUT     0E6H        ;THEN OFF
                IN      0E4H        ;READ STATUS BYTE
                ANI     002H        ;MASK ORDY2
                CPI     002H        ;TRUE OR FALSE
                JNZ     ORDY2       ;IS ORDY2 ?
;
;*************************************************************
;*                                                           *
;*  IS INPUT BUFFER READY ? (IE. IS INPUT BUFFER FULL?).... *
;*                                                           *
;*************************************************************
;
    IRDY1:      MVI     A,002H      ;PULSE RSTR CMD
                OUT     0E6H        ;TO CONTROL PORT
                MVI     A,000H      ;TURN OFF
                OUT     0E6H        ;RSTR PULSE
                IN      0E4H        ;READ STATUS BYTE
                ANI     001H        ;MASK IRDY 1
                CPI     001H        ;
                JNZ     IRDY1       ;IS IRDY1 ?
;
;*************************************************************
;*                                                           *
;*  COMPARE RECEIVED CMD BYTE WITH ORIG. PATTERN TO VERIFY  *
;*  CORRECTNESS. ERROR MSG  1 OUTPUT IF INCORRECT...        *
;*                                                           *
;*************************************************************
;
                MVI     A,004H      ;TURN DATA PULSE ON
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,006H      ;RSTR &DATA PULSE
```

```
          OUT     0E6H       ;TO CONT. PT.
          MVI     A,004H     ;TURN OFF RSTR ONLY
          OUT     0E6H       ;TO CONT. PT.
          NOP                ;TIME DELAY
          MVI     A,000H     ;TURN OFF DATA LINE
          OUT     0E6H
          IN      0E4H       ;LOAD CMD BYTE INTO ACCUM.
          ANI     019H       ;MASK CMD TO VERIFY
          CPI     019H       ;TO ORIGINAL PATTERN
          JNZ     ERRMSG1    ;WRITE OUT ERROR MSG. 1
;
;***************************************************************
;*                                                             *
;* WRITE PARAMETER BYTE  1 CONTAINING HEAD ADD. AND OUTPUT     *
;* TO DATA PORT...                                             *
;*                                                             *
;***************************************************************
;
;
PRAM1:    MVI     A,000H     ;OUTPUT PARAMETER BYTE
          OUT     0E4H       ; 1 WITH HEAD ADDRESS ZERO
          MVI     A,000H     ;TURN ACK/ ON
          OUT     0E5H       ;PORT B
          MVI     A,004H     ;TURN ON DATA LINE
          OUT     0E6H       ;TO CONT. PT.
          NOP                ;TIME DELAY
          MVI     A,005H     ;PULSE WSTR & DATA
          OUT     0E6H       ;OUT TO CONTROL PORT
          MVI     A,004H     ;TURN OFF WSTR ONLY
          OUT     0E6H       ;TO CONT. PT.
          NOP                ;TIME DELAY
          MVI     A,000H     ;ON THEN OFF...
          OUT     0E6H
          MVI     A,004H     ;RESTORE ACK/
          OUT     0E5H       ;PORT B
;
;
;***************************************************************
;*                                                             *
;* COMMAND VERIFY FOR FIRST PARAMETER BYTE...                  *
;*                                                             *
;***************************************************************
;
;
ORDY3:    MVI     A,002H     ;RSTR CMD TO
          OUT     0E6H       ;COMMAND PORT
          MVI     A,000H     ;ON AND
          OUT     0E6H       ;THEN OFF...
          IN      0E4H       ;READ STATUS BYTE
          ANI     002H       ;MASK ORDY3
          CPI     002H       ;TRUE OR FALSE
          JNZ     ORDY3      ;IS ORDY3 ?...
;
;***************************************************************
;*                                                             *
;* IS THE DISK CONTROLLER READY FOR INPUT (IRDY) ?......       *
;*                                                             *
;***************************************************************
;
;
IRDY2:    MVI     A,002H     ;PULSE RSTR CMD
          OUT     0E6H       ;TO CONTROL PORT
          MVI     A,000H     ;TURN OFF
          OUT     0E6H       ;RSTR PULSE
          IN      0E4H       ;READ STATUS BYTE
```

```
                ANI     001H        ;MASK IRDY2
                CPI     001H        ;
                JNZ     IRDY2       ;IS IRDY2?...
;
;****************************************************************
;*                                                              *
;*  READ BACK PARAMETER BYTE  1 FOR VERIFICATION....            *
;*                                                              *
;****************************************************************
;
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,006H      ;RSTR & DATA PULSE
                OUT     0E6H        ;READ BACK PRAM1
                MVI     A,004H      ;TURN OFF RSTR ONLY
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,000H      ;TURN OFF STROBE
                IN      0E4H        ;LOAD PRAM1 INTO ACCUM..
                ANA     A           ;MASK PRAM1 TO VERIFY
                JNZ     ERRMSG2     ;WRITE OUT ERROR MSG 2.
                DCR     B           ;IS THIS THE
                MOV     A,B         ;LAST PARAMETER BYTE ?...
                CPI     000H        ;IF SO
                JZ      GOBYTE      ;GO TO GO BYTE
                MOV     B,A         ;OTHERWISE SAVE PRAM COUNT.
;
;****************************************************************
;*                                                              *
;*  WRITE PARAMETER BYTE  2 CONTAINING LSB OF CYL. ADD.         *
;*  AND OUTPUT TO DATA PORT. IF Pram 2 IS IN OVERFLOW COND-     *
;*  TION CARRY OVER TO Pram. BYTE 3 AND CONTINUE TO VERIFY.     *
;*                                                              *
;****************************************************************
;
                LXI     H,PRAM2     ;LOAD ADD. PRAM2 IN HL REGS.
                MOV     A,M         ;MOVE PRAM VALUE TO ACCUM...
                OUT     0E4H        ;PUT PRAM2 TO OUTPUT PORT
                MOV     D,A
                PUSH    D           ;SAVE PRAM2 VALUE
                MVI     A,000H      ;TURN ACK/ ON
                OUT     0E5H         PORT B
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,005H      ;PULSE ON THEN OFF
                OUT     0E6H        ;DATA AND
                MVI     A,004H      ;TURN OFF WSTR ONLY
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,000H      ;WSTR TO
                OUT     0E6H        ;CONTROL PORT
                MVI     A,004H      ;RESTORE ACK/
                OUT     0E5H         PORT B
;
;****************************************************************
;*                                                              *
;*  COMMAND VERIFY FOR SECOND PARAMETER BYTE....                *
;*                                                              *
;****************************************************************
;
     ORDY4:     MVI     A,002H      ;RSTR CMD. TO
```

```
                CUT   0E6H          ;COMMAND PORT.
                MVI   A,000H        ;PULSE ON AND
                OUT   0E6H          ;THEN OFF...
                IN    0E4H          ;READ STATUS BYTE
                ANI   002H          ;MASK ORDY 4
                CPI   002H          ;TRUE OR FALSE
                JNZ   ORDY4         ;IS ORDY4 ?...
;
;*****************************************************************
;*                                                               *
;* IS THE DISK CONTROLLER READY FOR INPUT (IRDY) ?....           *
;*                                                               *
;*****************************************************************
;
   IRDY3:       MVI   A,002H        ;PULSE RSTR CMD
                OUT   0E6H          ;TO CONTROL PORT
                MVI   A,000H        ;TURN OFF
                CUT   0E6H          ;RSTR PULSE
                IN    0E4H          ;READ IN STATUS BYTE
                ANI   001H          ;MASK IRDY 3
                CPI   001H          ;TRUE OR FALSE
                JNZ   IRDY3         ;IS IRDY 3 ?....
;
;*****************************************************************
;*                                                               *
;* READ BACK PARAMETER BYTE  2 FOR VERIFICATION......            *
;*                                                               *
;*****************************************************************
;
                MVI   A,004H        ;TURN ON DATA LINE
                OUT   0E6H          ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI   A,006H        ;RSTR & DATA PULSE
                OUT   0E6H          ;READ BACK PRAM2
                MVI   A,004H        ;TURN OFF RSTR ONLY
                OUT   0E6H          ;TC CONT. PT.
                NOP                 ;TIME DELAY
                MVI   A,000H        ;TURN OFF STROBE
                OUT   0E6H
                IN    0E4H          ;LOAD PRAM2 INTO ACCUM..
;
;*****************************************************************
;*COMPARE Pram. EYTE 2 WITH ORIGINAL VALUE AND INCREMENT         *
;* TRACK NUMBER OR IF CHECK FAILS SEND ERROR MSG.  3...          *
;*                                                               *
;*****************************************************************
;
                POP   D             ;GET PRAM2 VALUE
                CMP   D             ;COMPARE TO ORIGINAL
                JNZ   ERRMSG3       ;SEND ERROR MSG.  3
                INR   A             ;CHG. TO NEXT TRACK  ...
                JZ    CARRY         ;IF C FLAG SET GO TO CARRY ROUT
                MOV   M,A           ;OTHERWISE STORE PRAM2....
                DCR   B             ;IS THIS THE
                MOV   A,B           ;LAST PARAMETER BYTE ?...
                CPI   000H          ;IF SO
                JZ    GOBYTE        ;GO TO GO BYTE..
                MOV   B,A           ;OTHERWISE SAVE PRAM COUNT.
                JMP   CONTINUE
```

```
;*************************************************************
;*                                                           *
;* THE CARRY ROUTINE IS UTILIZED IF Pram. BYTE 2 IS IN       *
;* overFLOW CONDITION TO INCREMENT PARAMETER BYTE  3...      *
;*                                                           *
;*************************************************************
;
;
  CARRY:        LXI     H,PRAM3      ;INCREMENT ADD. TO
                MOV     A,M          ;PRAM3 AND MOV TO ACCUM.
                INR     A            ;INCREMENT PRAM3
                CPI     002H         ;IS THIS CYLINDER  579 ?..
                JZ      EXIT         ;IF SO END INTVFY..
                MOV     M,A          ;SAVE NEW PRAM3 VALUE...
;
;*************************************************************
;*                                                           *
;* THE CONTINUE ROUTINE IS USED TO CONTINUE WITH VERIFY Cmd.*
;* THAT IS TRANSMIT PARAMETER BYTE  3...                     *
;*                                                           *
;*************************************************************
;
;
   CONTINUE:    LXI     H,PRAM3      ;LOAD ADD. PRAM3 IN HL REGS.
                MOV     A,M          ;MOVE PRAM3 VALUE TO ACCUM.
                OUT     0E4H         ;PUT PRAM3 TO OUTPUT PORT
                MOV     D,A
                PUSH    D            ;SAVE PRAM3 VALUE
                MVI     A,000H       ;TURN ACK/ ON
                OUT     0E5H          PORT B
                MVI     A,004H       ;TURN ON DATA LINE
                OUT     0E6H         ;TO CONT. PT.
                NOP                  ;TIME DELAY
                MVI     A,005H       ;PULSE ON THEN OFF
                OUT     0E6H         ;THE DATA AND
                MVI     A,004H       ;TURN OFF WSTR ONLY
                OUT     0E6H         ;TO CONT. PT.
                NOP                  ;TIME DELAY
                MVI     A,000H       ;WSTR TO
                OUT     0E6H         ;CONTROL PORT
                MVI     A,004H       ;RESTORE ACK/
                OUT     0E5H          PORT B
;
;*************************************************************
;*                                                           *
;* COMMAND VERIFY FOR THIRD PARAMETER BYTE......             *
;*                                                           *
;*************************************************************
;
;
   ORDY5:       mVI     A,002H       ;RSTR CMD. TO
                OUT     0E6H         ;COMMAND PORT
                MVI     A,000H       ;PULSE ON
                OUT     0E6H         ;THEN OFF....
                IN      0E4H         ;READ STATUS BYTE
                ANI     002H         ;MASK ORDY5
                CPI     002H         ;TRUE OR FALSE
                JNZ     ORDY5        ; IS ORDY5 ?...
;
;*************************************************************
;*                                                           *
;* IS DISK CONTROLLER READY FOR INPUT (IRDY4) ?...           *
;*                                                           *
;*************************************************************
;
```

```
;
    IRDY4:      MVI     A,002H      ;PULSE RSTR CMD.
                CUT     0E6H        ;TO CONTROL PORT
                MVI     A,000H      ;TURN OFF
                OUT     0E6H        ;RSTR PULSE
                IN      0E4H        ;READ IN STATUS BYTE
                ANI     001H        ;MASK IRDY4
                CPI     001H        ;TRUE OR FALSE
                JNZ     IRDY4       ;IS IRDY 4 ?..
;
;
;*****************************************************************
;*                                                               *
;*  READ BACK PARAMETER BYTE  3 FOR VERIFICATION...              *
;*                                                               *
;*****************************************************************
;
;
;
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,006H      ;RSTR & DATA PULSE
                OUT     0E6H        ;READ BACK PRAM3
                MVI     A,004H      ;TURN OFF RSTR ONLY
                CUT     0E6H        ;TC CONT. PT.
                NCP                 ;TIME DELAY
                MVI     A,000H      ;TURN OFF STROBE
                OUT     0E6H
                IN      0E4H        ;LOAD PRAM3 IN ACCUM..
;
;
;*****************************************************************
;*                                                               *
;*  COMPARE PRAM. BYTE 3 WITH ORIGINAL PATTERN AND CONTINUE *
;*, TO PARAMETER BYTE  4 OR TRANSMIT ERROR MSG.  4...            *
;*                                                               *
;*****************************************************************
;
;
;
                POP     D           ;PUT PRAM3 IN D REGS.
                CMP     D           ;COMPARE TO ORIGINAL
                JNZ     ERRMSG4     ;SEND ERROR MSG.  4..
                MOV     M,A         ;SAVE PRAM3
                DCR     B           ;IS THIS THE
                MOV     A,B         ;LAST PARAMETER BYTE ?..
                CPI     000H        ;IF SO
                JZ      GOBYTE      ;GO TO BYTE..
                MOV     B,A         ;OTHERWISE SAVE PRAM COUNT
;
;
;*****************************************************************
;*                                                               *
;*  WRITE PARAMETER BYTE  4 CONTAINING STARTING SECTOR ADD. *
;*  FOR INITIALIZATION THIS WILL ALWAYS BE ZERO...               *
;*                                                               *
;*****************************************************************
;
;
;
                MVI     A,000H      ;LOAD PRAM4 (START SECT.)
                CUT     0E4H        ;LOGICAL SECTOR ZERO
                MVI     A,000H      ;TURN ACK/ ON
                OUT     0E5H         PORT B
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NCP                 ;TIME DELAY
                MVI     A,005H      ;PULSE WSTR & DATA
                CUT     0E6H        ;TO COMMAND PORT
                MVI     A,004H      ;TURN OFF WSTR ONLY
                CUT     0E6H        ;TO CONT. PT.
```

```
          NOP                 ;TIME DELAY
          MVI     A,000H      ;TURN CFF PULSE TO
          CUT     0E6H        ;COMMAND PORT
          MVI     A,004H      ;RESTORE ACK/
          CUT     0E5H         PORT B
;
;
;*************************************************************
;*                                                           *
;*  START CMD. VERIFICATION OF PARAMETER BYTE 4 BY READING   *
;*  STATUS BYTE...                                           *
;*                                                           *
;*************************************************************
;
;
    ORDY6:  MVI     A,002H      ;RSTR CMD. TO
            CUT     0E6H        ;COMMAND PORT
            MVI     A,000H      ;PULSE ON
            OUT     0E6H        ;THEN OFF...
            IN      0E4H        ;READ IN STATUS BYTE
            ANI     002H        ;MASK ORDY6
            CPI     002H        ;TRUE OR FALSE
            JNZ     ORDY6       ;IS ORDY6 ?...
;
;*************************************************************
;*                                                           *
;*  IS DISK CONTRCLLER READY FOR INPUT (IRDY5) ?...          *
;*                                                           *
;*************************************************************
;
;
    IRDY5:  MVI     A,002H      ;PULSE RSTR CMD.
            OUT     0E6H        ;TO CONTROL PORT
            MVI     A,000H      ;TURN OFF
            OUT     0E6H        ;RSTR PULSE
            IN      0E4H        ;READ IN STATUS BYTE
            ANI     001H        ;MASK IRDY5
            CPI     001H        ;TRUE OR FALSE
            JNZ     IRDY5       ;IS IRDY5 ?...
;
;*************************************************************
;*                                                           *
;*  READ BACK PARAMETER BYTE  4 FOR VERIFICATION.....        *
;*                                                           *
;*************************************************************
;
;
            MVI     A,004H      ;TURN ON DATA LINE
            OUT     0E6H        ;TO CONT. PT.
            NOP                 ;TIME DELAY
            MVI     A,006H      ;RSTR & DATA PULSE
            OUT     0E6H        ;READ BACK PRAM4
            MVI     A,004H      ;TURN OFF RSTR ONLY
            CUT     0E6H        ;TO CONT. PT.
            NOP                 ;TIME DELAY
            MVI     A,000H      ;TURN OFF
            CUT     0E6H        ;STROBE...
            IN      0E4H        ;LOAD PRAM4 IN
;
;*************************************************************
;*                                                           *
;*  COMPARE PARAMETER BYTE 4 WITH ORIG. PATTERN AND CONTINUE*
;*  TO PARAMETER BYTE  5 OR TRANSMIT ERROR MSG.  5...        *
;*                                                           *
;*************************************************************
;
```

```
        ANA     A           ;COMPARE TO ORIG. VALUE
        JNZ     ERRMSG5     ;PRINT ERROR MSG.
        DCR     B           ;IS THIS THE
        MOV     A,B         ;LAST PARAMETER BYTE
        CPI     000H        ;IF SO
        JZ      GOBYTE      ;GO TO GOBYTE
        MOV     B,A         ;OTHERWISE SAVE PRAM COUNT
;
;*****************************************************************
;*                                                               *
;*  WRITE PARAMETER BYTE  5 CONTAINING # OF SECTORS TO BE        *
;*  PROCESSED. FOR INITIALIZATION OF A COMPLETE TRACK A TIME*
;*  THIS NUMBER WILL ALWAYS BE 00....                            *
;*                                                               *
;*****************************************************************
;
        MVI     A,000H      ;LOAD PRAM5
        OUT     0E4H        ;23D OUTPUT PORT
        MVI     A,000H      ;TURN ACK/ ON
        OUT     0E5H        ;PORT B
        MVI     A,004H      ;TURN ON WSTR
        OUT     0E6H        ;TO CONT. PT.
        NOP                 ;TIME DELAY
        MVI     A,005H      ;PULSE WSTR AND DATA
        OUT     0E6H        ;TO COMMAND PORT
        MVI     A,004H      ;TURN OFF WSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        NCP                 ;TIME DELAY
        MVI     A,000H      ;TURN OFF PULSE
        CUT     0E6H        ;TO CMD. PORT
        MVI     A,004H      ;RESTORE ACK/
        CUT     0E5H        ;PORT B
;
;*****************************************************************
;*                                                               *
;*  START CMD. VERIFICATION OF PARAMETER BYTE 5 BY READING       *
;*  STATUS BYTE...                                               *
;*                                                               *
;*****************************************************************
;
ORDY7:  MVI     A,002H      ;RSTR CMD TO
        CUT     0E6H        ;CMD. PORT
        MVI     A,000H      ;PULSE ON
        OUT     0E6H        ;THEN OFF...
        IN      0E4H        ;READ IN STAT.BYTE
        ANI     002H        ;MASK ORDY7
        CPI     002H        ;TRUE OR FALSE
        JNZ     ORDY7       ;IS ORDY7 ?..
;
;*****************************************************************
;*                                                               *
;*  IS DISK CCNTRCLLER READY FOR INPUT ?.....                    *
;*                                                               *
;*****************************************************************
;
IRDY6:  MVI     A,002H      ;PULSE RSTR CMD.
        OUT     0E6H        ;TO CONTROL PORT
        MVI     A,000H      ;TURN OFF
        OUT     0E6H        ;RSTR PULSE
        IN      0E4H        ;READ STAT. BYTE
        ANI     001H        ;MASK IRDY6
        CPI     001H        ;TRUE FALSE
        JNZ     IRDY6       ;IS IRDY6 ?
```

```
;
;
;*****************************************************************
;*                                                               *
;*  READ BACK PARAMETER BYTE  5 FOR VERIFICATION...             *
;*                                                               *
;*****************************************************************
;
;
;
                MVI     A,004H          ;TURN ON DATA LINE
                OUT     0E6H            ;TO CONT. PT.
                NOP                     ;TIME DELAY
                MVI     A,006H          ;RSTR & DATA PULSE
                OUT     0E6H            ;READ BACK PRAM5
                MVI     A,004H          ;TURN OFF RSTR ONLY
                OUT     0E6H            ;TO CONT. PT.
                NCP                     ;TIME DELAY
                MVI     A,000H          ;TURN OFF
                OUT     0E6H            ;STROBE...
                IN      0E4H            ;LOAD PRAM5 ACCUM.
;
;
;*****************************************************************
;*                            *
;*  COMPARE PARAMETER BYTE 5 WITH ORIG. PATTERN AND CONTINUE*
;*  TO PARAMETER BYTE  6 OR XMIT ERROR MSG  6..                 *
;*                                                               *
;*****************************************************************
;
;
;
            ANA     A               ;COMPARE TO
            JNZ     ERRMSG6         ;PRINT ERR. MSG.
            DCR     B               ;IS THIS THE
            MOV     A,B             ;LAST PRAM BYTE
            CPI     000H            ;IF SO
            JZ      GOBYTE          ;
            MOV     B,A             ;SAVE PRAM CT.
;
;
;*****************************************************************
;*                                                               *
;*  WRITE PRAM. BYTE 6 CONTAINING NORMAL/SPARED TRACK AND        *
;*  DEFFECTIVE SECTOR ADDRESS. NORMALLY NOT USED BUT MUST        *
;*  BE XMITTED ANYWAY. CONTAINS ALL ZEROS.......                 *
;*                                                               *
;*****************************************************************
;
;
;
                MVI     A,024H          ;LOAD PRAM6
                OUT     0E4H            ;TO OUTPUT PORT
                MVI     A,000H          ;TURN ACK/ ON
                OUT     0E5H             PORT B
                MVI     A,004H          ;TURN ON WSTR
                CUT     0E6H            ;TO CONT. PT.
                NOP                     ;TIME DELAY
                MVI     A,005H          ;PULSE WSTR AND
                CUT     0E6H            ;DATA CMD. PT.
                MVI     A,004H          ;TURN OFF WSTR ONLY
                OUT     0E6H            ;TO CONT. PT.
                NOP                     ;TIME DELAY
                MVI     A,000H          ;TURN OFF PULSE
                OUT     0E6H            ;CMD PORT
                MVI     A,004H          ;TURN ACK/ ON
                OUT     0E5H             PORT B
```

```
;
;*************************************************************
;*                                                           *
;*  COMMENCE COMMAND VERIFICATION OF PARAMETER BYTE 6 BY     *
;*  READING STATUS BYTE...                                   *
;*                                                           *
;*************************************************************
;
;
     ORDY8:     MVI     A,002H      ;RSTR CMD TO
                OUT     0E6H        ;CMD. PORT
                MVI     A,000H      ;PULSE CN
                OUT     0E6H        ;THEN OFF..
                IN      0E4H        ;READ STAT. BYTE
                ANI     002H        ;MASK ORDY8
                CPI     002H        ;TRUE OR FALSE
                JNZ     ORDY8       ;IS ORDY8 ? ...
;
;*************************************************************
;*                                                           *
;*  IS DISK CONTROLLER READY FOR INPUT (IRDY7) ?...          *
;*                                                           *
;*************************************************************
;
;
     IRDY7:     MVI     A,002H      ;PULSE RSTR CMD.
                OUT     0E6H        ;TO CONT. PT.
                MVI     A,000H      ;TURN OFF
                OUT     0E6H        ;RSTR PULSE
                IN      0E4H        ;READ STATUS BYTE
                ANI     001H        ;MASK IRDY7
                CPI     001H        ;TRUE OR FALSE
                JNZ     IRDY7       ;IS IRDY7 ?
;
;*************************************************************
;*                                                           *
;*  READ BACK PARAMETER BYTE  6 FOR VERIFICATION...          *
;*                                                           *
;*************************************************************
;
;
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,006H      ;RSTR & DATA PULSE
                OUT     0E6H        ;READ BACK PRAM6
                MVI     A,004H      ;TURN OFF RSTR ONLY
                OUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,000H      ;TURN OFF
                OUT     0E6H        ;STROBE
                IN      0E4H        ;LOAD PRAM6 IN ACCUM.
;
;*************************************************************
;*                                                           *
;*  COMPARE PARAMETER BYTE  6 WITH ORIGINAL PATTERN AND      *
;*  CONTINUE TO GO BYTE FOR EXECUTION OF INITIALIZATION...   *
;*                                                           *
;*************************************************************
;
                CPI     024H        ;COMPARE TO ORIG. VALUE
                JNZ     ERRMSG7     ;PRINT ERROR MSG. 7
                DCR     B           ;IS THIS THE
                MOV     A,B         ;LAST PRAM BYTE ?
                CPI     000H        ;IF SO
```

```
                JZ      GOBYTE     ;GO TO GOBYTE
;
;***************************************************************
;*                                                             *
;* OUTPUT GO BYTE TO DISK CONTROLLER. THE GO BYTE CAUSES THE*
;* COMMAND TO BE EXECUTED AND MAY CONTAIN ANY VALUE. FOR      *
;* SIMPLICITY THE GO BYTE WILL BE FFH....                     *
;*                                                             *
;***************************************************************
;
;
    GOBYTE:     MVI     A,0FFH     ;LOAD GO BYTE
                OUT     0E4H       ;TO OUTPUT PORT
                MVI     A,000H     ;TURN ACK/ ON
                OUT     0E5H       ;PORT B
                MVI     A,004H     ;TURN ON WSTR
                CUT     0E6H       ;TO CONT. PT.
                NOP                ;TIME DELAY
                MVI     A,005H     ;PULSE WSTR AND
                OUT     0E6H       ;DATA TO CMD. PT.
                MVI     A,004H     ;TURN OFF WSTR ONLY
                OUT     0E6H       ;TO CONT. PT.
                NOP                ;TIME DELAY
                MVI     A,000H     ;TURN OFF PULSE
                OUT     0E6H       ;TO CMD. PT.
                MVI     A,004H     ;RESTORE ACK/
                OUT     0E5H       ;PORT B
;
;***************************************************************
;*                                                             *
;* THE WAIT STATUS MODULE IS USED TO READ BACK TERMINATION *
;* BYTE. TERMINATION STATUS IS ACCESSED BY READING FROM      *
;* THE CONTROLLER DATA PORT IN RESPONSE TO ATTN TRUE, USING*
;* THE TERMINATION PROTOCOL....                               *
;*                                                             *
;***************************************************************
;
;
    CBUSY2:     MVI     A,002H     ;RSTR CMD.
                OUT     0E6H       ;TO CONT. PT.
                MVI     A,000H     ;STROBE OFF..
                OUT     0E6H       ;
                IN      0E4H       ;READ STATUS BYTE
                ANI     010H       ;IS CBUSY TRUE ?
                CPI     010H       ; OR FALSE ?
                JNZ     CBUSY2     ;CONTRCLLER BUSY GO BACK.
;
;***************************************************************
;*                                                             *
;* READ STATUS BYTE TO SEE IF INPUT BUFFER READY ? (IRDY8).*
;*                                                             *
;***************************************************************
;
;
    IRDY8:      MVI     A,002H     ;PULSE RSTR CMD.
                OUT     0E6H       ;TC CONT. PT.
                MVI     A,000H     ;TURN OFF
                OUT     0E6H       ;RSTR PULSE
                IN      0E4H       ;READ STATUS BYTE
                ANI     001H       ;MASK IRDY8
                CPI     001H       ;
                JNZ     IRDY8      ;IS IRDY8 ?
```

```
;
;*************************************************************
;*                                                           *
;* READ FROM CONTROLLER DATA PORT THE TERM. STATUS BYTE.     *
;* DETERMINE ERRCR CONDITIONS THAT HAVE OCCURRED DURING THE*
;* COMMAND EXECUTION. THE CODE IN BITS 0-3 INDICATE REASON *
;* FOR TERMINATICN....                                       *
;*                                                           *
;*************************************************************
;
;
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT. PT.
                NCP                 ;TIME DELAY
                MVI     A,006H      ;PULSE RSTR & DATA
                OUT     0E6H        ;TO CONT. PT.
                MVI     A,004H      ;TURN CFF RSTR ONLY
                CUT     0E6H        ;TO CONT. PT.
                NOP                 ;TIME DELAY
                MVI     A,000H      ;TURN OFF
                OUT     0E6H        ;RSTR & DATA..
                IN      0E4H        ;READ IN TER. STAT. BYTE
                ANA     A           ;MASK TER. STAT. FOR ERROR
                JNZ     ERRMSG8     ;PRINT ERROR MSG.
                JMP     INTVFY      ;RETURN TO BEGINNING
;
;
ERRMSG1:        LXI     H,ERROR1
                MVI     B,35D
START1:         MOV     D,M
                caLL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START1
;
;
ERRMSG2:        LXI     H,ERROR2
                MVI     B,33D
START2:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START2
;
;
ERRMSG3:        LXI     H,ERROR3
                MVI     B,33D
START3:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START3
;
;
ERRMSG4:        LXI     H,ERROR4
                MVI     B,33D
START4:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START4
;
;
ERRMSG5:        LXI     H,ERROR5
                MVI     B,33D
```

```
START5:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START5
;
ERRMSG6:        LXI     H,ERROR6
                MVI     B,33D
START6:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START6
;
ERRMSG7:        LXI     H,ERROR7
                MVI     B,33D
START7:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START7
;
ERRMSG8:        LXI     H,ERROR8
                MVI     B,34D
START8:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START8
;
EXIT:           NOP
                NOP
                LXI     H,COMP
                MVI     B,33D
START9:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      FINISH
                INX     H
                JMP     START9
;
CONOUT:         IN      0EDH
                ANI     00000001B
                CPI     00000001B
                JNZ     CONOUT
                MOV     A,D
                OUT     0ECH
                RET
;
PRAM2:  DB  0
PRAM3:  DB  0
ERROR1: DB 'COMMAND BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR2: DB 'PRAM1 BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR3: DB 'PRAM2 BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR4: DB 'PRAM3 BYTE RECEIVED IN ERROR...',0DH,0AH
;
```

```
ERROR5: DB 'PRAM4 BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR6: DB 'PRAM5 BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR7: DB 'PRAM6 BYTE RECEIVED IN ERROR...',0DH,0AH
;
ERROR8: DB 'TERMINATION STATUS BYTE ERROR...',0DH,0AH
;
COMP:   DB 'THIS COMPLETES INTVFY OF DISK...',0DH,0AH
;
;
;
 FINISH:         NOP
                 NOP
                 RST 1          ;END INTVFY PGM.
```

APPENDIX B

```
 ORG    3000H
;
;*****************************************************************
;*                                                               *
;*  THIS IS A SUBRCUTINE TO EITHER READ OR WRITE TO THE          *
;*  MICROPOLIS 1223-1 HARD DISK DEPENDING ON THE COMMAND         *
;*  BYTE. IF THE COMMAND BYTE IS A 4EH IT IS A READ COM-         *
;*  MAND OR 47H FOR A WRITE COMMAND.....                         *
;*                                                               *
;*****************************************************************
;
        NOP
        NOP
;
;*****************************************************************
;*                                                               *
;*  INITIALIZE CONDITICNS BY PROGRAMING 8255 TO MODE 2,          *
;*  CLEARING ACCUMULATOR, SETTING ACK/ ON AND SETTING            *
;*  PARAMETER COUNTER TO 6...                                    *
;*                                                               *
;*****************************************************************
;
                MVI     A,001H      ;SET DISK CONTROLLER
                OUT     0E5H        ;ENABLE  PT.B
 READRITE:      SUB  A              ;CLEAR ACCUM.
                ADI     6           ;LOAD PRAM CT.
                MOV     B,A
                MVI     A,0C0H      ;PGM. 8255
                CUT     0E7H        ;MODE TWO
                MVI     A,005H      ;TURN ON ACK/
                CUT     0E5H         PORT B
;
;*****************************************************************
;*                                                               *
;*  READ STATUS BYTE AND MASK TO SEE IF DISK CONTROLLER          *
;*  IS NOT BUSY (CBUSY=1). FOLLOW BY CHECKING TO SEE IF          *
;*  HOST MAY SEND COMMAND BYTE (ORDY=1)....                      *
;*                                                               *
;*****************************************************************
;
        CALL    CBUSY     ;IS CONT. BUSY?
        CALL    ORDY      ;READY FOR CMD?
;
;*****************************************************************
;*                                                               *
;*  LOAD READ CR WRITE COMMAND BYTE INTO DISK CONTROL PORT.*
;*                                                               *
;*****************************************************************
;
        LXI     H,CMD     ;ADD. OF CMD BYTE
        MOV     A,M       ;MOVE CMD TO ACCUM.
        OUT     0E4H      ;PUT CMD TO OUT PT.
        MOV     D,A       ;SAVE ORIG.
        PUSH    D         ;CMD. BYTE
        MVI     A,001H    ;TURN ACK/ ON
        OUT     0E5H       PORT B
        MVI     A,001H    ;STROBE RSTR ON
```

```
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;WSTR OFF
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;RESTORE ACK/ ON
        OUT     0E5H        PORT B
;
;*****************************************************************
;*                                                               *
;* COMMENCE COMMAND VERIFICATION BY READING STATUS BYTE          *
;* FOR ORDY & IRDY...                                            *
;*                                                               *
;*****************************************************************
;
        CALL    ORDY        ;READY FOR CMD?
        CALL    IRDY        ;CMD READY FOR READ BACK
;
;*****************************************************************
;*                                                               *
;* COMPARE RECEIVED COMMAND BTYE WITH ORIGINAL PATTERN           *
;* TO VERIFY CORRECTNESS...                                      *
;*                                                               *
;*****************************************************************
;
        MVI     A,004H      ;TURN DATA PULSE ON
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,006H      ;TURN ON RSTR & DATA
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF RSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN OFF DATA
        OUT     0E6H        ;TO CONT. PT.
        IN      0E4H        ;READ IN CMD.
        POP     D           ;LOAD ORIG. IN D
        CMP     D           ;COMPARE COMMANDS
        JNZ     ERRMSG1     ;FAIL? PRINT MSG 1
        PUSH    D           ;SAVE FOR RD WR DECISION
;
;*****************************************************************
;*                                                               *
;* WRITE PARAMETER BYTE  1 CONTAINING HEAD ADDRESS AND           *
;* OUTPUT TO DATA PORT...                                        *
;*                                                               *
;*****************************************************************
;
        LXI     H,PRAM1     ;LOAD ADD. OF HEAD
        MOV     A,M         ;PRAM1 VALUE TO ACCUM.
        OUT     0E4H        ;OUTPUT PRAM1 TO DATA PT.
        MOV     D,A         ;SAVE PRAM1 BYTE
        PUSH    D           ;ON STACK
        MVI     A,001H      ;TURN ACK/ ON
        OUT     0E5H        PORT B
        MVI     A,004H      ;TURN ON DATA LINE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;STROBE WSTR ON
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF WSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN OFF DATA LINE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;RESTORE ACK
        OUT     0E5H        PORT B...
;
```

```
;*************************************************************
;*                                                           *
;* COMMENCE PARAMETER  1 VERIFICATION BY READING STATUS      *
;* BTYE FOR ORDY & IRDY..                                    *
;*                                                           *
;*************************************************************
;
;
        CALL    ORDY        ;HOST MAY SEND PRAM2
        CALL    IRDY        ;PRAM READY TO READ BACK
;
;*************************************************************
;*                                                           *
;* READ BACK PARAMETER BYTE  1 FOR VERIFICATION...           *
;*                                                           *
;*************************************************************
;
;
        MVI     A,004H      ;TURN ON DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,006H      ;TURN ON RSTR & DATA
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF RSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN OFF DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        IN      0E4H        ;LOAD PRAM1 ACCUM.
        POP     D           ;LOAD ORIG. PRAM1 IN D
        CMP     D           ;COMPARE PRAMS
        JNZ     ERRMSG2     ;FAILED? PRINT MSG.
        DCR     B           ;DCR PRAM COUNT
        JZ     GOBYTE       ;LAST PRAM END PGM.
;
;*************************************************************
;*                                                           *
;* WRITE PARAMETER BYTE 2 CONTAINING LSB OF CYLINDER         *
;* ADDRESS AND OUTPUT TO DATA PORT. IF PARAMETER  2 IS       *
;* IN OVERFLOW CONDITION CARRY OVER TO PARAMETER BYTE  3     *
;* AND CONTINUE WITH EXECUTION...                            *
;*                                                           *
;*************************************************************
;
;
        LXI     H,PRAM2     ;LOAD ADD. PRAM2
        MOV     A,M         ;MOVE PRAM2 VALUE
        OUT     0E4H        ;TO OUTPUT PORT
        MOV     D,A         ;SAVE PRAM2
        PUSH    D           ;VALUE IN STK.
        MVI     A,001H      ;TURN ACK  ON
        OUT     0E5H         PORT B
        MVI     A,004H      ;TURN ON DATA LINE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;TURN ON DATA & WSTR
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF WSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN OFF DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;RESTORE ACK
        OUT     0E5H        PORT B
;
;*************************************************************
;*                                                           *
;* COMMENCE COMMAND VERIFY FOR PARAMETER BYTE 2...           *
;*                                                           *
;*************************************************************
```

```
        CALL    ORDY        ;HOST MAY SEND PRAM 3
        CALL    IRDY        ;PRAM2 READY TO READ BACK
;
;*************************************************************
;*                                                           *
;*  READ BACK PARAMETER BYTE 2 FOR VERIFICATION...           *
;*                                                           *
;*************************************************************
;
;
        MVI     A,004H      ;TURN ON DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,006H      ;RSTR & DATA ON
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF RSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN DATA OFF
        OUT     0E6H        ;TO CONT. PT.
        IN      0E4H         ;LOAD IN PRAM2
;
;*************************************************************
;*                                                           *
;*  COMPARE PARAMETER BYTE 2 WITH ORIGINAL VALUE. IF         *
;*  CHECK FAILS PRINT ERROR MSG.  3...                       *
;*                                                           *
;*************************************************************
;
;
        POP     D           ;GET PRAM2 VALUE
        CMP     D           ;COMPARE TO ORIG.
        JNZ     ERRMSG3     ;FAILED? SEND MSG.
        INR     A           ;CHG. TO NEXT TRACK
        JC      CARRY       ;OVERFLOW TO PRAM  4
        MOV     M,A         ;OTHERWISE SAVE PRAM2
        DCR     B           ;IS THIS LAST PRAM?
        JZ      GOBYTE      ;IF SO END PGM.
        JMP     CONTINUE    ;OTHERWISE CONTINUE
;
;*************************************************************
;*                                                           *
;*  THE CARRY ROUTINE IS UTILIZED IF PARAMETER BYTE  2 IS    *
;*  IN OVERFLOW CONDITION TO INCREMENT PARAMETER BYTE  3..   *
;*                                                           *
;*************************************************************
;
;
   CARRY:       LXI     H,PRAM3     ;LOAD ADD. PRAM3
                MOV     A,M         ;PRAM VALUE TO ACCUM.
                INR     A           ;INCREMENT PRAM3
                CPI     002H        ;IS THIS TRACK  579
                JZ      EXIT        ;IF SO END PGM.
                MOV     M,A         ;SAVE NEW PRAM3 BYTE
;
;*************************************************************
;*                                                           *
;*  THE CONTINUE ROUTINE IS USED TO CONTINUE WITH READ       *
;*  WRITE COMMAND. THAT IS TRANSMIT PARAMETER BYTE  3        *
;*                                                           *
;*************************************************************
;
;
     CONTINUE:     LXI     H,PRAM3   ;LOAD ADD. PRAM3
                   MOV     A,M       ;MOVE PRAM3 VALUE
```

```
        OUT     0E4H        ;PUT PRAM3 OUTPUT PT.
        MOV     D,A         ;SAVE PRAM  3
        PUSH    D           ;BYTE
        MVI     A,001H      ;TURN ACK  ON
        OUT     0E5H        ;PORT B
        MVI     A,004H      ;TURN ON DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;TURN ON WSTR & DATA
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF WSTR ONLY
        OUT     0E6H        ;TO CONT. PT,
        MVI     A,000H      ;TURN OFF DATA STROBE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,005H      ;RESTORE ACK
        OUT     0E5H        ;PORT B
;
;**********************************************************
;*                                                        *
;* COMMENCE VERIFICATION OF PARAMETER BYTE  3....         *
;*                                                        *
;**********************************************************
;
;
        CALL    CRDY        ;HOST MAY SEND PRAM  4
        CALL    IRDY        ;PRAM3 READY FOR READ BACK
;
;**********************************************************
;*                                                        *
;* READ BACK PARAMETER BYTE  3 FOR VERIFICATION...        *
;*                                                        *
;**********************************************************
;
;
        MVI     A,004H      ;TURN ON DATA LINE
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,006H      ;TURN ON RSTR & DATA
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,004H      ;TURN OFF RSTR ONLY
        OUT     0E6H        ;TO CONT. PT.
        MVI     A,000H      ;TURN OFF DATA LINE
        OUT     0E6H        ;TO CONT. PT.
        IN      0E4H        ;LOAD IN PRAM3
;
;**********************************************************
;*                                                        *
;* COMPARE PARAMETER BYTE  3 WITH ORIGINAL PATTERN AND    *
;* CONTINUE TO PARAMETER BYTE  4 OR PRINT ERROR MSG.  4   *
;* IF CHECK FAILS...                                      *
;*                                                        *
;**********************************************************
;
;
        POP     D           ;PUT ORIG, PRAM3 IN D
        CMP     D           ;COMPARE TWO VALUES
        JNZ     ERRMSG4     ;FAILED? PRINT MSG.
        MOV     M,A         ;SAVE PRAM  3 VALUE
        DCR     B           ;DCR PRAM COUNT
        JZ      GOBYTE      ;IS THIS THE LAST PRAM?
;
;**********************************************************
;*                                                        *
;* WRITE PARAMETER BYTE  4 CONTAINING STARTING SECTOR     *
;* ADDRESS...                                             *
;*                                                        *
;**********************************************************
```

```
        LXI     H,PRAM4    ;LOAD ADD. PRAM4
        MOV     A,M        ;PUT PRAM4 VALUE IN ACCUM.
        OUT     0E4H       ;PRAM4 OUT TO OUTPUT PT.
        MOV     D,A        ;SAVE PRAM4
        PUSH    D          ;BYTE
        MVI     A,001H     ;TURN ACK  ON
        OUT     0E5H        PORT B
        MVI     A,004H     ;TURN ON DATA LINE
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,005H     ;PULSE WSTR & DATA ON
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,004H     ;TURN OFF WSTR ONLY
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,000H     ;TURN OFF DATA LINE
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,005H     ;RESTORE ACK
        OUT     0E5H        PORT B
;
;*********************************************************
;*                                                       *
;* COMMENCE VERIFICATION OF PARAMETER BYTE  4 BY READING *
;* STATUS BYTE FOR ORDY & IRDY...                        *
;*                                                       *
;*********************************************************
;
        CALL    ORDY       ;HOST MAY SEND PRAM  5
        CALL    IRDY       ;PRAM4 READY TO READ BACK
;
;*********************************************************
;*                                                       *
;* READ BACK PARAMETER BYTE  4 FOR VERIFICATION.......   *
;*                                                       *
;*********************************************************
;
        MVI     A,004H     ;TURN ON DATA LINE
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,006H     ;TURN RSTR & DATA ON
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,004H     ;TURN OFF RSTR ONLY
        OUT     0E6H       ;TO CONT. PT.
        MVI     A,000H     ;TURN OFF DATA
        OUT     0E6H       ;TO CONT. PT.
        IN      0E4H       ;READ IN PRAM4
;
;*********************************************************
;*                                                       *
;* COMPARE PARAMETER BYTE  4 WITH ORIGINAL PATTERN AND   *
;* CONTINUE TO PARAMETER BYTE  5 OR IF FAIL PRINT MSG 5. *
;*                                                       *
;*********************************************************
;
        POP     D          ;RECALL ORIG. PRAM
        CMP     D          ;COMPARE TWO VALUES
        JNZ     ERRMSG5    ;FAILED? PRINT MSG.
        DCR     B          ;DETERMINE PRAM CT.
        JZ      GOBYTE     ;IF LAST PRAM
```

```
;
;***************************************************************
;*                                                             *
;* WRITE PARAMETER BYTE  5 CONTAINING NUMBER OF SECTORS        *
;* TO BE PROCESSED. FOR THIS PARTICULAR CASE THIS SYSTEM       *
;* WILL ONLY WRITE OR READ ONE SECTOR AT A TIME THERE-         *
;* FORE PRAM5=001H.................                             *
;*                                                             *
;***************************************************************
;
;
        MVI     A,001H    ;LOAD PRAM5
        OUT     0E4H      ;TO OUTPUT PT.
        MVI     A,001H    ;TURN ACK  ON
        OUT     0E5H      PORT B
        MVI     A,004H    ;TURN ON DATA LINE
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,005H    ;WSTR & DATA ON
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,004H    ;TURN OFF WSTR ONLY
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,000H    ;TURN OFF DATA LINE
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,005H    ;RESTORE ACK
        OUT     0E5H       PORT B
;
;
;***************************************************************
;*                                                             *
;* COMMENCE VERIFICATION OF PARAMETER BYTE  5 BY READING       *
;* STATUS BYTE.....                                            *
;*                                                             *
;***************************************************************
;
;
        CALL     ORDY     ;HOST MAY SEND PRAM6
        CALL     IRDY     ;PRAM READY TO READ BACK
;
;
;***************************************************************
;*                                                             *
;* READ BACK PARAMETER BYTE  5 FOR VERIFICATION.....           *
;*                                                             *
;***************************************************************
;
;
        MVI     A,004H    ;TURN ON DATA LINE
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,006H    ;RSTR & DATA ON
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,004H    ;TURN OFF RSTR ONLY
        OUT     0E6H      ;TO CONT. PT.
        MVI     A,000H    ;TURN OFF DATA LINE
        OUT     0E6H      ;TO CONT. PT.
        IN      0E4H      ;READ IN PRAM5
;
;
;***************************************************************
;*                                                             *
;* COMPARE PARAMETER BYTE  5 WITH ORIGINAL PATTERN AND         *
;* CONTINUE TO PARAMETER BTYE  6 OR IF FAILED PRINT ERROR      *
;* MSG  6.......                                               *
;*                                                             *
;***************************************************************
;
;
        ANI     001H      ;COMPARE
        CPI     001H      ;TO ORIG. VALUE
        JNZ     ERRMSG6   ;FAIL? PRINT MSG  6
```

```
        DCR     B        ;DCR PRAM CT.
        JZ      GOBYTE   ;LAST PRAM EXECUTE PGM.
;
;****************************************************************
;*                                                              *
;*  WRITE PARAMETER BYTE  6 CONTAINING NORMAL/SPARED TRACK      *
;*  AND DEFFECTIVE SECTOR ADDRESS. NORMALLY NOT USED BUT        *
;*  MUST BE TRANSMITTED ANYWAY. CONTAINS ALL ZEROS......        *
;*                                                              *
;****************************************************************
;
;
        MVI     A,025H   ;LOAD PRAM6
        OUT     0E4H     ;TO OUTPUT PT.
        MVI     A,001H   ;TURN ACK  ON
        OUT     0E5H      PORT B
        MVI     A,004H   ;TURN ON WSTR
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,005H   ;TURN WSTR & DATA ON
        OUT     0E6H     ;TO CONT. PT.
        MVI     A,004H   ;TURN WSTR OFF
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,000H   ;TURN OFF DATA
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,005H   ;RESTORE ACK
        OUT     0E5H      PORT B
;
;****************************************************************
;*                                                              *
;*  COMMENCE VERIFICATION OF PARAMETER BYTE  6 BY READING       *
;*  STATUS BYTE FCR ORDY & IRDY.......                          *
;*                                                              *
;****************************************************************
;
;
        CALL     ORDY    ;HOST MAY SEND GC BYTE
        CALL     IRDY    ;PRAM6 READY TO SEND BACK
;
;****************************************************************
;*                                                              *
;*  READ BACK PARAMETER BYTE  6 FOR VERIFICATION....            *
;*                                                              *
;****************************************************************
;
;
        MVI     A,004H   ;TURN ON DATA LINE
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,006H   ;TURN DATA RSTR ON
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,004H   ;TURN OFF RSTR ONLY
        OUT     0E6H     ;TO CONT.PT.
        MVI     A,000H   ;TURN OFF DATA LINE
        OUT     0E6H     ;TO CONT.PT.
        IN      0E4H     ;READ IN PRAM6
;
;****************************************************************
;*                                                              *
;*  COMPARE PARAMETER BYTE  6 WITH ORIGINAL PATTERN AND         *
;*  CONTINUE TO GO BYTE OR IF FAIL PRINT ERROR MSG.  7...       *
;*                                                              *
;****************************************************************
;
;
        CPI     025H     ;CCMPARE TO 24
        JNZ     ERRMSG7  ;FAILED? PRINT MSG.
```

```
        DCR     B           ;IS THIS LAST
        JZ      GCBYTE      ;YES GO TO EXECUTION
;
;**************************************************************
;*                                                            *
;*  OUTPUT GO BYTE TO DISK CONTROLLER. THE GO BYTE CAUSES     *
;*  THE COMMAND TO BE EXECUTED AND MAY CONTAIN ANY VALUE.     *
;*  FOR SIMPLICITY THE GO BYTE WILL BE FFH......              *
;*                                                            *
;**************************************************************
;
;
GOBYTE:         MVI     A,0FFH      ;LOAD GO BYTE
                OUT     0E4H        ;TO DATA PORT
                MVI     A,001H      ;TURN ACK  ON
                OUT     0E5H         PORT B
                MVI     A,004H      ;TURN DATA PULSE ON
                OUT     0E6H        ;TO CONT. PT.
                MVI     A,005H      ;TURN ON WSTR
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,004H      ;TURN WSTR OFF
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,000H      ;TURN DATA OFF
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,005H      ;RESTORE ACK
                OUT     0E5H         PORT B
;
;**************************************************************
;*                                                            *
;*  THE DECISION ROUTINE DETERMINES IF THE COMMAND WAS A      *
;*  READ OR WRITE. IT THEN PASSES CONTROL OVER TO THE AP-     *
;*  PROPRIATE MODULE FOR EXECUTION.....                       *
;*                                                            *
;**************************************************************
;
;
        POP     D           ;RECOVER CMD. BYTE
        MOV     A,D         ;MOV CMD. TO ACCUM.
        CPI     047H        ;TEST FOR WRITE CMD.
        JZ      WRITE       ;GO TO WR. MODULE
;
;**************************************************************
;*                                                            *
;*  THE READ MODULE READS THE STATUS BYTE AND DETERMINES      *
;*  IF DATA IS REQUSTED OR ATTENTION IS TRUE. IF SO DATA      *
;*  PORT PUTS BYTE TO DATA BUFFER. (FOR DEMONSTRATION THIS    *
;*  DATA WILL BE PRINTED TO MONITOR CRT..)                    *
;*                                                            *
;**************************************************************
;
;
            LXI     B,200H
            LXI     H,TABLE1
READ:       CALL    STATUS      ;IS CONTROLLER BUSY?
             MVI     A,004H     ;TURN ON DATA LINE
             OUT     0E6H       ;TO CONT. PT.
             MVI     A,006H     ;TURN ON RSTR & DATA
             OUT     0E6H       ;TO CONT.PT.
             MVI     A,004H     ;TURN RSTR OFF
             OUT     0E6H       ;TO CONT.PT.
             MVI     A,000H     ;TURN OFF DATA LINE
             OUT     0E6H       ;TO CONT.PT.
             IN      0E4H        ;READ IN DATA BYTE
             MOV     M,A
             MOV     D,A
             CALL    CONOUT
```

```
            INX     H
            DCR     C
            JNZ     READ
            DCR     B
            JZ      EXIT
            JMP     READ        ;GET NEXT BYTE
;
;*************************************************************
;*                                                           *
;*  THE WRITE MODULE READS STATUS BYTE AND DETERMINES IF     *
;*  IF DATA REQUEST OR ATTENTION BITS ARE SET TO ONE. IF     *
;*  SO GETS DATA FROM BUFFER AND WRITES TO DATA PORT....     *
;*                                                           *
;*************************************************************
;
;
;
    WRITE:      CALL    STATUS      ;IS CBUSY?
                LXI     H,PROG      ;LOAD PGM ADD
                MOV     A,M         ;NEXT BYTE
                OUT     0E4H        ;OUT PGM BYTE
                MVI     A,001H      ;TURN ACK  ON
                OUT     0E5H         PORT B
                MVI     A,004H      ;TURN ON DATA LINE
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,005H      ;TURN WSTR & DATA ON
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,004H      ;TURN OFF WSTR ONLY
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,000H      ;TURN OFF DATA LINE
                OUT     0E6H        ;TO CONT.PT.
                MVI     A,005H      ;RESTORE ACK
                OUT     0E5H         PORT B
                INR     M           ;ADD. NEXT BYTE
                JMP     WRITE       ;GET NEXT BYTE
;
;*************************************************************
;*                                                           *
;*  THE WAIT STATUS MODULE IS USED TO READ BACK TERMINAT-    *
;*  ION BYTE. THIS BYTE IS ACCESSED BY READING FROM THE      *
;*  CONTROLLER DATA PORT IN RESPONSE TO ATTN=1,USING TER-    *
;*  MINATION PROTOCOL.......                                 *
;*                                                           *
;*************************************************************
;
;
    WAIT:   MVI     B,00CH
            LXI     H,TABLE
    WAIT1:  CALL    CBUSY       ;IS CONTROLLER BUSY?
            CALL    IRDY        ;TER. STAT. BYTE READY?
            MVI     A,004H      ;TURN ON DATA LINE
            OUT     0E6H        ;TO CONT.PT.
            MVI     A,006H      ;TURN ON RSTR & DATA
            OUT     0E6H        ;TO CONT.PT.
            MVI     A,004H      ;TURN OFF RSTR ONLY
            OUT     0E6H        ;TO CONT.PT.
            MVI     A,000H      ;TURN OFF DATA
            OUT     0E6H        ;TO CONT.PT.
            IN      0E4H        ;READ IN TER.STAT.BYTE
            MOV     M,A
            INX     H
            DCR     B
            JZ      EXIT
            JMP     WAIT1
            NOP
            NOP
            RST     1           ;TO CALLING PROGRAM
```

```
;
;****************************************************************
;*                                                              *
;* THE CONTROLLER BUSY SUBRCUTINE IS USED TO DETERMINE IF      *
;* THE DISK CONTROLLER IS BUSY (CBUSY=0). IF CONTROLLER        *
;* IS NOT BUSY CONTROL IS RETURNED TO MAIN PROGRAM AND         *
;* EXECUTION CONTINUES......                                    *
;*                                                              *
;****************************************************************
;
;
CBUSY:      PUSH    B           ;SAVE
            PUSH    D           ;THE CONTENTS
            PUSH    H           ;OF ALL
            PUSH    PSW         ;REGISTERS
CBUSY1:     MVI     A,002H      ;RSTR ON
            OUT     0E6H        ;TO CONT. PORT
            MVI     A,000H      ;RSTR OFF
            OUT     0E6H        ;TO CONT. PT.
            IN      0E4H        ;READ STATUS WORD
            ANI     010H        ;DOES CBUSY=1
            CPI     010H        ;OR NOT
            JNZ     CBUSY1      ;GO BACK IF SO
            POP     PSW
            POP     H
            POP     D
            POP     B
            RET
;
;
;****************************************************************
;*                                                              *
;* THE OUTPUT READY SUBROUTINE IS USED TO DETERMINE IF         *
;* THE DISK CONTRCLLER IS READY TO RECEIVE A WORD FROM         *
;* THE HOST COMPUTER........                                    *
;*                                                              *
;****************************************************************
;
;
ORDY:       PUSH    B           ;SAVE THE
            PUSH    D           ;CONTENTS
            PUSH    H           ;OF ALL
            PUSH    PSW         ;REGISTERS
ORDY1:      MVI     A,002H      ;RSTR ON
            OUT     0E6H        ;TO CONT.PT.
            MVI     A,000H      ;RSTR OFF
            OUT     0E6H        ;TO CONT. PT.
            IN      0E4H        ;READ STATUS WORD
            ANI     002H        ;DOES ORDY=1?
            CPI     002H        ;TRUE OR FALSE?
            JNZ     ORDY1       ;BUFFER EMPTY
            POP     PSW
            POP     H
            POP     D
            POP     B
            RET
;
;
;****************************************************************
;*                                                              *
;* THE INPUT READY SUBROUTINE IS TO DETERMINE IF THE DISK      *
;* CONTROLLER HAS A BYTE READY TO BE INPUT TO THE HOST...      *
;*                                                              *
;****************************************************************
;
;
IRDY:       PUSH    B           ;SAVE THE
            PUSH    D           ;CONTENTS
            PUSH    H           ;OF ALL
```

```
        PUSH    PSW     ;REGISTERS
IRDY1:  MVI     A,002H  ;TURN RSTR ON
        OUT     0E6H    ;TO CONT. PT.
        MVI     A,000H  ;TURN RSTR OFF
        OUT     0E6H    ;TO CONT. PT.
        IN      0E4H    ;READ IN STATUS BYTE
        ANI     001H    ;MASK IRDY
        CPI     001H    ;FOR IRDY=1
        JNZ     IRDY1   ;IF NOT READY GO BACK
        POP     PSW
        POP     H
        POP     D
        POP     B
        RET
;
;
;**************************************************************
;*                                                            *
;* THE STATUS SUBROUTINE DETERMINES IF ATTENTION OR DATA      *
;* REQUEST BITS ARE SET...                                    *
;*                                                            *
;**************************************************************
;
;
;
STATUS:     PUSH    B
            PUSH    D
            PUSH    H
            PUSH    PSW
STATUS1:    MVI     A,002H  ;RSTR ON
            OUT     0E6H    ;TO CONT.PT.
            MVI     A,000H  ;RSTR OFF
            OUT     0E6H    ;TO CONT.PT.
            IN      0E4H    ;READ STAT. BYTE
            MOV     D,A     ;SAVE BYTE
            MVI     A,080H  ;LOAD MASK
            ANA     D       ;PERFORM TEST
            CPI     080H
            JZ      WAIT    ;IF TRUE GO TO WAIT
            MVI     A,020H  ;LOAD MASK
            ANA     D
            JNZ     STATUS1 ;GO BACK CTHERWISE
            POP     PSW
            POP     H
            PCP     D
            POP     B
            RET
;
;
;
ERRMSG1:    LXI     H,ERROR1
            MVI     B,35D
START1:     MOV     D,M
            CALL    CONOUT
            DCR     B
            JZ      EXIT
            INX     H
            JMP     START1
;
;
ERRMSG2:    LXI     H,ERROR2
            MVI     B,33D
START2:     MOV     D,M
            CALL    CONOUT
            DCR     B
            JZ      EXIT
            INX     H
            JMP     START2
;
;
```

```
ERRMSG3:        LXI     H,ERROR3
                MVI     B,33D
START3:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START3
:
ERRMSG4:        LXI     H,ERROR4
                MVI     B,33D
START4:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START4
:
ERRMSG5:        LXI     H,ERROR5
                MVI     B,33D
START5:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START5
:
ERRMSG6:        LXI     H,ERROR6
                MVI     B,33D
START6:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START6
:
ERRMSG7:        LXI     H,ERROR7
                MVI     B,33D
START7:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START7
:
ERRMSG8:        LXI     H,ERROR8
                MVI     B,34D
START8:         MOV     D,M
                CALL    CONOUT
                DCR     B
                JZ      EXIT
                INX     H
                JMP     START8
:
EXIT:            NOP
                 NOP
                 LXI     H,COMP
                 MVI     B,34D
START9:          MOV     D,M
                 CALL    CONOUT
                 DCR     B
                 JZ      FINISH
                 INX     H
```

```
                  JMP     START9
;
;
    CONOUT: IN       0EDH
            ANI      00000001B
            CPI      00000001B
            JNZ      CONOUT
            MOV      A,D
            OUT      0ECH
            RET
;
;
    CMD:    DB        00EH
;
    PRAM1:  DB        00H
;
    PRAM2:  DB        00H
;
    PRAM3:  DB        00H
;
    PRAM4:  DB        00H
;
;
    ERROR1: DB 'COMMAND BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR2: DB 'PRAM1 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR3: DB 'PRAM2 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR4: DB 'PRAM3 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR5: DB 'PRAM4 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR6: DB 'PRAM5 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR7: DB 'PRAM6 BYTE RECEIVED IN ERROR...',0DH,0AH
;
    ERROR8: DB 'TERMINATION STATUS BYTE ERROR...',0DH,0AH
;
    COMP:   DB 'THIS COMPLETES RD/WR CMD........',0DH,0AH
;
    TABLE:     DS     12
    TABLE1:    DS     512
    PROG:      DB     00
;
;
;**********************************************************
;*                                                        *
;* THE FINISH ROUTINE PROVIDES A PRCGRAMMED RESET FOR     *
;* THE DISK CONTRCLLER WHICH AUTOMATICALLY INITIALIZES    *
;* THE CONTROLLER AFTER EXECUTION OF EACH COMMAND...      *
;*                                                        *
;**********************************************************
;
;
    FINISH:    MVI    A,000H   ;PULSE ENABLE
               OUT    0E5H     ;ON PT. B
               MVI    A,001H   ;TURN OFF ENABLE
               OUT    0E5H      PORT B
               NOP
               RST    1
```

LIST OF REFERENCES

1. Tanenbaum, A.S., Computer Networks, Prentice Hall, 1981

2. Burton,T., 1223 Rigid Disk Drive Reference Manual, Rev. B, March, 1981

BIBLIOGRAPHY

Burton, T., Micropolis Installation Guidelines Manual, November 1980

Dempsey, J. A., Basic Digital Electronics with MSI Applications, Addison Wesley, 1979

Leventhal, L. A., Introduction to Microprocessors: Software, Hardware, Programming, Prentice-Hall, 1978

Intel 8080 Microcomputer Systems User's Guide, Intel Corp., September, 1975

Intel OEM Computers, Intel Corp., November, 1977

Intellec Hardware Reference Manual, Intel Corp., 1967

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
2.	Department Chairman, Code 62 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
3.	Dr. M. L. Cotton, Code 62Cc Department of Electrical Engineering Naval Postgraduate School Montrery, California 93940	2
4.	Dr. Rudy Panholzer, Code 62Pz Deparment of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
5.	Dr. Uno R. Kodres, Code 52Kr Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
6.	LT. William H. Brown, USN 386A Bergin Dr. Montrery, Califirnia 93940	1

A design of a
hard disk interface
for the micropolis
1223-1.

197[illegible]57

Thesis
B823445 Brown
c.1

A design of a
hard disk interface
for the micropolis
1223-1.